ARTIFICIAL LIFE AS A VEHICLE FOR ANOMALIES DETECTION ON INDUSTRIAL CONTROL SYSTEM

THE BEHAVIOUR OF BIRD SWARMS AND HOW IT CAN BE APPLIED IN ICS

MICHAEL OKEKE

A submission presented in partial fulfilment of the requirements of the University of South Wales/ Prifysgol De Cymru for the degree of Doctor of Philosopher

March 2018

UNIVERSITY OF SOUTH WALES
Pontypridd, UK

ABSTRACT

The proliferation of attacks on critical infrastructures in recent time has posed questions on how to secure such systems. Industrial Control Systems (ICS) such as Supervisory Control and Data Acquisition (SCADA) is used on critical infrastructures such as manufacturing industries, nuclear sites, oil and gas industries, locomotives and among others. These systems generate a lot of data and the current detection engine cannot handle such data. This project is a demonstration of an innovative idea titled *"Artificial Life as a Vehicle in Detecting Malicious Behaviours on Industrial Control Systems".* This provides model framework for detecting malicious activities on Industrial Control System (ICS). The project provides artificial life model for securing ICS. The model is based on the behaviours of swarm of birds. Swarm or flock of birds have some characteristics that worth emulating such as their approach in detecting predator in their environment. These animals are not necessarily very intelligent animal but their approach in group for the detection and avoidance of predator was studied and adopted. Hence, detection in this respect is adopting the flock of bird's approach in detecting predator. The important findings of this project are their individual or single bird action during flight that made them forms group as well as their information transfer from one bird to the entire flock. These are the two vital properties of the flock of birds that enhances their detection of predator. These approaches were modelled for the detection of anomalies on ICS. The model proved that it is possible to apply this approach on ICS and the architecture shows that the model can detect unknown anomalies and handle big data challenges.

Acknowledgement

Firstly, I would like to thank God for His guidance, wisdom and love bestowed upon me throughout this period. His love brings nourishment to my spirit, soul and body which gave me the boldness and dominion to surmount all the adversities faced in every stage of this project. I would like to thank my able supervisor, professor Andrew Blyth for your invaluable support and for believing in me, despite your tight schedules, you always find time for our meetings. Your soft approach to situations and problems made me believe that I can do more which turns my concerns into passion. I am indeed very grateful and would like to say thank you. My gratitude goes to my colleagues and all the members of ISRG or cybersecurity both present and past for their advice and contribution to the success of this study, you have inspired me positively, especially Peter Eden and Mathew Evans for allowing me to use their equipment in the lab. Special thanks to the members of the research office especially Linos Spargo for her ever willing to assist.

My heart and gratitude go to my parents, late Chief Ephraim Okeke and Mrs Patricia Okeke for the way they brought me up. You always told me that whatever I put my heart upon to achieve, is possible, that delay is never denial. Thank you for imparting this mindset in me. I will like to thank my lovely Wife, Jana Okeke and my two wonderful children Calvin and Marvin for their patience and always there for me, you are an inspiration and a dream come true. You made this a success. My gratitude goes to all the members of Believers Love World Cardiff for your prayers and support. I would like to acknowledge all my office mates Dr. Oteng Tabone and Madina Mukta for your support for such a time.

Table of Contents

List of Figures

List of Tables

List of Algorithms

Code Pseudocode Listing in Appendix

List of Abbreviations

ABC - Artificial Bee Colony

AC – Control Information

ACCM – Ant Colony Clustering Model

ACO – Ant Colony Optimisation

ADU – Application Data Unit

AIS – Artificial Immune System

AL – Artificial Life

ALF – Artificial Life Framework

ANN – Artificial Neural Network

APCI – Application Protocol Control Information

API – Application Interface

ARPANET – Advanced Research Projects Administration

ASCII – American Standard Code for Information Interchange

ASDU – Application Service Data Unit

BNN – Bayesian Neural Network

CCTV – Close Circuit TV

COTP – Connection Oriented Transport Protocol

CPN - Coloured Petri Nets

CPU – Central Processing Unit

CRC – Cyclic Redundancy Check

DA – Decision Algorithm

DCS – Distributed Control System

DDoS – Distributed Denial of Service

DER – Distributed Electric Resources

DIDS – Distributed Intrusion Detection Systems

DLL – Dynamic Link Library

DNP3 – Distributed Network Protocol 3

DR – Design Research

DSR – Design Science Research

DTI – Decision Tree Induction

EIA/TIA – Electronic Industries Association/ Telecommunication Industries Association

EPA – Enhanced Performance Architecture

FCFS – First Come First Serve

FINER – Feasible Interesting Novel Ethical Relevant

GDA – Global Detection Agent

GNPT – General Network Protocol Agent

HDFS – Hadoop Distributed File System

HLC – High Level Control

HMI – Human Machine Interface

HMM – Hidden Markov Model

ICS – Industrial Control System

IDS – Intrusion Detection System

IEC – International Electrical Commission

IoT – Internet of Things

ISO – International Standard Organisation

ISS – Internet Security System

KNN – Kohonen Neural Network

LAN – Local Area Network

LDA – Local Detection Agent

LLC – Low Level Control

LRC – Longitudinal Redundancy Check

LSB – Less Significant Bit

MAS – Malti Agent System

MDAP – Modbus Application Protocol

MDP – Markov Decision Process

MFM – Movable Fixed Object

MITM – Man In The Middle

ML – Machine Learning

MSB – Most Significant Bit

MTU – Master Terminal Unit

NIDS – Network Intrusion Detection System

NIST – National Institute of Standards and Technology

NNM – Neural Network Model

NTZ No-Trust-Zone

OPC – Open Platform Communication

OSI – Open System Interconnection

PDU – Protocol Data Unit

PLC – Programmable Logic Controller

PSO – Particle Swarm Optimisation

QN – Queueing Network

RS – Recommended Standard

RTT – Round Trip-delay Time

RTU – Remote terminal Unit

SANN – Simulated Annealing Neural Network

SANS – System Administration, Networking and Security

SAP – Service Access Point

SCADA – Supervisory Control and Data Acquisition

SCPT – Specific SCADA Protocol Traffic

SI – Swarm Intelligent

SOM – Self Organising Map

SVM – Support Vector Machine

TIA – Totally Integrated Automation

TPKT – Transport Packet

TTL – Time-To-Live

UML – Unified Modelling Languag

CHAPTER I

1 INTRODUCTION

This chapter will deal with the introduction of Industrial Control Systems (ICS) and Artificial Life. The introduction will usher the reader into some basic nomenclatures that will aid in understanding the project. This chapter highlights some of the problems that are facing ICS that requires solution, which is the reason for adopting flocks of birds' approach to predator for anomalies detection on ICS. Thus, this chapter will introduce the artificial life and all the properties that will be needed for its application on ICS.

ICS and Supervisory Control And Data Acquisition (SCADA) in particular have evolved from single, monolithic entities to the Internet of Things (IoT) (McClanahan, 2002). SCADA is used in many different critical infrastructures, from nuclear management to utility, power, waste and engine management on ships and planes. Security of ICS is paramount as deviation from normal operation may result in putting lives at risk. The technological revolution in recent time to name a few such as Internet, protocols interoperability among manufacturers and cloud computing has redefined the operation and security of SCADA systems. The interconnection of these devices in a distributed environment through the Internet and other means exposes them to various attacks (Gosine, 2017; Yang, and Zhao, 2014).

This project proposed the use of artificial life for the detection of anomalies on the ICS. The approach is adopting the flock of bird's approach to predator. Emulating the detection approach seen in the flock of birds equates to the artificial life in this project. Hence, when artificial life is being discussed in relation to this project, flocks of bird's behavior is the focus. This animal flies in groups of thousands and even millions. Their movements are well coordinated that it seems as if they are being remotely controlled. Their predator detection in such a large group is exceptional. These behaviors and other properties seen in these animals were modelled by emulating their detection approach for anomalies detection on ICS.

The detection engine will be placed in the cloud that will be designed for this purpose. Hadoop framework will be configured for data streaming using apache Spark and HBase as a database. The data from the sensors will be aggregated by the Ingester and streamed into the system. The Spark which has an API for the algorithm will pick the data for further analysis.

1.1 Problems Situations for ICS

The interconnectivity of different devices used in ICS forms an ecosystem of devices connected together. The possibility of malicious intent in such an ecosystem of interconnected devices such as ICS is high (Galvin, 2016). Managing and securing such an ecosystem can be daunting for security engineers and operators due to the dispersed nature of it. Most of these technologies are produced without security in mind and securing them takes process. Bruce Schneier, (Schneier, 2012, chapter 16) pointed out that sometimes it seems as if the attackers have the upper hand due to the fact that the technological advancement is faster than security. This is true, as security personnel need to study the new technology before developing new methods, models and approaches of securing it. Some of the perpetrators of this attacks are not doing it only for financial gains as many thinks. John Gordineer the director of product marketing for network security at Dell wrote;

"*They are trying to overwhelm the SCADA system and cause a denial of service, what they are trying to do is not to steal data but to shut the devices down. We hypothesized that there is less of a financial motive here than a disruption of service type of motive*" (Dell, 2015).

There have been series of attacks on critical infrastructures over the years since the revelation of the Stuxnet in 2010 as reported by Ball (2017). The rate of increase is phenomenal which are obvious from the incidences caused by these attacks. Report from the Security Intelligence as reported by McMillen (2016) stated that attacks targeting ICS increased by 110% in 2016. It is amazing that this report also indicated United States of America as the highest source followed by Pakistan and China. The highest attacked nation was also USA followed by China and Israel. Countries like USA received the highest number of attacks because they have the highest number of connected ICS online (Coats, 2017). The problem is that the attack vectors are changing as technology advances. In 2013, the New York Bowman Dam was hacked into by the group reportedly from Iran as reported by the Department of Justice (2016). According to the report, these hackers gained unauthorised access to the Dam which gave them access to the water level regulation. However, during this time, the regulator function was manually switched off. This would have caused a disaster for the state, if they have allowed these hackers to play with those functions.

In 2015, the Ukrainians suffered power outage in some of their cities as a result of cyber-attack as reported in ICS-CERT (2016) and by McMillen (2016). Spear phishing email was used in penetrating the power company's network. This gave them access to the network and the

malware was installed into the system. The malware exploited the vulnerability in Microsoft excel for their own advantage. The BlackEnergy 3 malware as it was named is a bot that started with version 1 and now version 3, commonly written as BE3. The BE3 that was used for the Ukrainian energy company exploited the macros on the excel and through that gained access to the system. This showed another possibility of causing physical damage to a system by a malware such as seen in the case of Stuxnet. One thing about the BlackEnergy 3 attack was that, it penetrated the system and took control of the SCADA system. The perpetrators flooded the telephone line and caused denial of service in order to prevent customers from reaching the provider. The question is how did all these happened? As earlier explained, the Microsoft excel office was used and the BlackEnergy 3 was embedded in it using social engineering. This office document was sent to the administrative department employee who opened it. By opening the document, the employee was prompted to enable macros which connected to the command and control of the attacker. This gave the attacker access to the system and they were able to carry out their reconnaissance and controlled the system remotely.

There have been reports in the past few years about train collisions in Europe and around the world as compiled and reported by Gupta, (2014). The movement of people now from place to place have increased, therefore the system must be stepped up to accommodate the growth. The present growth in the movement of people mounts pressure on the system with a lot of interconnections and responsibilities. In 2010, there was a collision in Belgium that killed up to 18 people (Cole, 2010). In the same 2010, there was a collision in India that killed 63 people and left 150 people injured (Gupta and Boral, 2010). Train collision was reported in many countries such as Tunisia, Indonesia, Ukraine, Estonia and among others as reported by BBC online. In 2017, there was a train collision in Egypt that left 40 people dead and 133 injured (Levenson, Arif, and Elsayed, 2017). This is just to mention a few to show that the issue of this kind of anomalies is still alive and well. Even in 2018 in February, two passengers train collided in Austria that left one person dead and 22 injured (Beer, 2018). These systems use SCADA system for monitoring the operation in the environment. The current anomalies showed that, the systems still have a long way to go in resolving such problems.

The simple course of these anomalies is the rate of interconnections of devices and technologies online, which are increasing (Rainie and Anderson, 2017). The means of attack is increasing that you don't really need to be a professional to learn how to hack. The sophistication of ICS attack vector in recent time can be seen in the way the Stuxnet penetrated the Iranian plant

without being notices for some years (Nourian and Madnick, 2018). The penetration was through the drivers DLL, which caused the system to shut down and damaged some of the components. Damages to this kind of system can cause outage as well as loss of money, reputation damage from client perspective and among others. The problem is that, securing the system from attack might not be easy. The current detection system might not be capable because of huge data produced by ICS. ICS generates enormous amount of data that makes it difficult for traditional Intrusion Detection System (IDS) to analyze (Zuech, Khoshgoftaar, Wald, 2015). Thus, the volume of data as well as the attack vectors is overwhelming the current detection mechanisms such as witnessed in the case of Stuxnet attack in 2010 (Langner, 2011). Big Data generators such as ICS, requires innovative technologies for anomalies detection as well as data processing and storage (Sungard, 2015).

1.2 Artificial Life Definition

This project proposes the use of Artificial Life (AL) as an Intrusion Detection System (IDS) for detecting anomalies on ICS. The intrusions here are the activities or behaviors that are deemed anomaly based on the outlined criteria according to Oxford dictionary. The real definition of life has occupied the mind of researchers for many years, thus the recent definition is based on the area of specialization. The notion on the existence of life and the characteristics of it have been defined based on the subject of specialization. Even in the area of specializations, there are still some misunderstandings of the real definition. These are based on both current and past scientific papers that was reviewed. Below are some of the definitions;

The word Artificial Life was first coined by Christopher Langton in the year 1986 and his definition of Artificial Life written as A-Life or life was that it is a study with regards to its emergence from inanimate object such as molecules (Langton C, 1986). Many scientists have problem with his definition of life and believes that there should be more to it. Below are some definitions of life or Artificial Life.

Based on the perspective of Smolin (1997 P.195), life is a self-organized none equilibrium system that is controlled through a symbolic program or simply controlled by a program with the possibilities of reproduction of itself. However, definition from Ackley (2014) was that life is anything that dynamically preserve pattern. He was of the opinion that computer programs can exhibit life like based on the characteristics. Programs should make autonomous

decisions and should not be deterministic in nature. Joyce (2009) also define life as a "*self-sustained chemical system that is capable of undergoing Darwinian evolution"*

From all the definitions above, it is easy to depict that each definition of life is based on the subject of specialization as earlier mentioned. Ackley was a computer scientist and his definition was related to computing. His definition will explain the application or where the artificial life will be applied. However, Smolin was a physicist and his definition and application of it was in that direction. The same goes to Joyce who is a chemist as well as Langton initial definition of Alife. Many scientists like Tu Xuyana believes that Langton definition of Alife was related only to engineering. He believes that the deficiency in the application of his definition was that it applies only to engineering (Xuyana, 2005).

The scientific definition of life based on the information above was that life evolves. Some definition was that it as an emergent property that has the ability to exhibit intelligent. Looking at the definitions and the explanations gave by these scientific works, below is the definition from this project.
This study defines Life as an emergent property that is created with a purpose, which exhibit intelligent and followed a certain pattern for its existence.

1.3 The Research Aim

The primary aim of this study is to explore, experiment and explain through the use of models the application of artificial life in detection of anomalies on ICS. ICS as a big data generator, the model should handle big data issue. This study will focus on modelling the collective behaviours seen in flocks of birds, such as detection of predator in such a large group.

1.3.1 Objectives

1. To carry out research on the possibility of using the flocks of bird's approach for anomalies detection on ICS.
2. Carry out research on the possible best way to model the approach for the detection of anomalies on ICS
3. Carry out research on the best possible solution for big data

4. To create a SCADA test rig upon which a predator prey model can execute and to create a predator prey model. The test rig will facilitate the collection of initial data from the PLC and HMI that will be fed into the model in order to determine the anomalies.
5. To create the models and evaluate it for performance in order to validate the hypothesis.
6. To compare and contrast the result in order to prove or disprove the assertion that a bird of prey model can enhance the detection of anomalies on Industrial Control Systems.

1.3.2 Hypotheses

"*Predator prey model approach can enhance the detection of anomalies on SCADA industrial process control systems"*

1.4 Contributions to Knowledge

Due to the recent development and technological advancements such as interconnectivity of the SCADA system to the internet, the system now faces new threats. The rate at which the threat is increasing is phenomenal. The fact being that these systems generate a lot of data which the current detection engine cannot handle. The primary contribution of this project is the possibility of using the proposed approach for the detection of anomalies on ICS. Its uniqueness lies in the distributed detection seen in the flocks of birds. Emulating this for the protection of SCADA system is a unique contribution to the body of knowledge. The second contribution is the architecture that can accommodate big data for ICS. The architecture is in such a way that data flows to the cloud and processed in the cloud.

1.5 Thesis Structure

This section of this work deals with the structure of the thesis. The thesis constitutes a total of eight chapters. Below are the brief summaries of each chapter for a head start.

Chapter I - Introduction

This chapter introduces some of the concept that are vital for this project to come to fruition, such as the ICS, SCADA and artificial life. It briefly highlights some of the challenges or problem in securing ICS.

Chapter II – Background

The second chapter which is the background will highlight some of the background information needed in order to follow the flow of the project. It shows the difference between ICS and

SCADA system and the generations of SCADA system till this present day. It went further in introducing some of the present challenges in securing SCADA system. Based on these current problems that were found, a hypothesis that captured the solution to the problem was drawn.

Chapter III – Literature Review

This chapter went further in reviewing both the current and past literatures in the area of securing SCADA system as well as solving the problem of big data. Based on the literature, it was evidence that SCADA system is one of the big data generator and current security system might not be able to handle it. Therefore, a new approach is needed that can detect and handle big data issues. Hence, the research went further into artificial life and in particular the approach seen in the flocks of birds. The viability of using this idea in detecting anomalies on SCADA system was research and a conclusion was reached.

Chapter IV – System Design and Model

The chapter provided the first design of the proposed system. The design helped in virtualising the proposed idea. This gave birth to required components of the system and eventually the first model.

Chapter V Model

This chapter introduces the second model of bird that will be used for the case study. The test rig in chapter IV was used in playing out the model. This test rig represents the SCADA system that facilitates data collection. CPN tool was used for petri nets modelling of the approach and different algorithms for different points were designed.

Chapter VI Case Study

This chapter introduces the case study that was designed in the previous chapter. The algorithm was directed to the case study and the results were tested against the hypothesis. The case study helped in locating different parts of the system that should be protected and how to apply the protection on them.

Chapter VII - Evaluation

This chapter is for the evaluation of the UML as well as the petri nets model of the case study. The UML model was evaluated using the QN model as well as the case study for performance. The end result showed that the model is fit for the purpose.

Chapter VIII – Conclusion

This is the concluding part of the thesis and it summed up both the aim and objectives of the study. The reflective review showed that the aim and objectives were met. Recommendation for further study on this model was given as well as the contribution this work has made to the body of knowledge.

1.6 Summary

This introductory chapter shades some light on the project aim and objectives at this point. It highlights and introduces the issues the current ICS is facing with regards to security. These issues are the reason for this PhD project which is to emulate the behavior of swarm or flock of birds in detecting predators. The behavior will be modelled for the detection of anomalies on ICS. Using the behavior of group of animals in solving complex problems is one of the research focus in recent time.

During the course of this project, four presentations and academic papers as well as journal were produced, see appendix. The first presentation was titled Artificial Life for securing ICS, which was presented in 2015 at the 3rd Annual postgraduate researchers presentation day. The first academic paper was published in International Journal of Mathematical, Computational, Physical, Electrical and Computer Engineering Volume 10. This article was titled "Adopting flocks of bird's approach to predator for anomalies detection on industrial control systems". The second paper was a conference paper titled "No-Trust-Zone architecture for securing supervisory control and data acquisition". The third paper was also a conference paper presented at IEEE NIGERCON which was titled "Emulating the distributed detection Approach in Flocks of Birds for Securing SCADA Systems"

CHAPTER II

2 BACKGROUND

This chapter will give some background information on the project, which will involve some of the equipment and terms associated with this project. It will help someone who is not familiar with some technical terms such as Industrial Control Systems, artificial intelligent, artificial life, multi agent systems and among others. These are some of the systems and technology this chapter in this project will be addressing.

Industrial Control System is an encompassing word that involve many components (Hayden, Assante, and Conway, 2014). However, the focus of this project or the environment for this project is the SCADA system. Understanding the genesis of such environment will be very vital and this chapter will delve into it. This will give some basic background knowledge of the components and the functions of the system as well as the product of the project. The product here is an artificial life model. This will introduce some other areas that are related to artificial life such as artificial intelligent and multi agent system. These areas are closely related to each other and some of these words are sometimes used interchangeably. However, they are not the same which will be evidence in this chapter.

2.1 Industrial Control Systems (ICS)

ICS has become a common word that is heard everywhere and from different people. The actual definition can be very crucial because of its role in industry and everyday life activities such as seen in automobiles, aeronautics, production industries, homes, and among others. Thus, the definition of the concept is very vital, as the understanding of it will develop the awareness of its important in everyday activities.

NIST (2014) defines ICS as a term involving the combination and coupling of different control systems in order to achieve a desired industrial result in an environment. This involves components such as measuring devices (Sensors, thermometers, gauges, etc), Field devices (Programmable Logic Controller (PLC), Remote Terminal Unit (RTU), etc), and Actuating components, Supervisory Control and Data Acquisition (SCADA), Distributed Control Systems (DCS). These components comprise of the ICS, but most industries regard and only recommend SCADA system as the only component of the ICS. The reason being that SCADA

is the widely known among the ICS and most widely researched on in recent time due to the proliferation of the attacks.

2.2 Supervisory Control and Data Acquisition (SCADA)

The age of technological revolution has seen increase in the way things are being done and controlled; from nuclear system to aerospace management, from locomotives to utility and water managements, from engine management on ships to internet of a thing such as social medias managements (Stouffer, Falco and Scarfone, 2008). The complexity in managing such an ecosystem in technological environment can be very demanding for engineers with regards to technological and operational standpoint. The idea of SCADA system for the control and monitoring such ecosystem ameliorated the difficulties in management of industrial system. Such industrial system can be seen nowadays, as mentioned above but not limited to; Aviation, astronomical environment, agriculture, atmospheric monitoring, building management, consumer product and entertainment, military environment, nuclear environment monitoring and among others (Babovic and Velagic, 2009).

The idea and technology behind SCADA has been in existence for over 30 years, Boyer (2009). SCADA system is made up of software's and hardware's built together for monitoring and controlling systems for efficiency and reliability measures. The sensors are placed on the remote sites that report the conditions of the sites through remote system. SCADA system has its own way of communication and transportation of information. There are few communication protocols used in SCADA system depending on the organization. Some organization has their custom-made protocol, while some use open protocol such as Modbus, Distributed Network Protocol version 3 (DNP3), ProfiNet, IEC 60870-5 -101/-103/104 and among others. Understanding these various components of SCADA will shed light on the areas that needs protection. The various subsystem of SCADA will be explained further in the SCADA architecture section below.

2.2.1 SCADA Architecture:

SCADA architecture for security is mostly based on the need of the organization as explained by Karnouskos and Colombo (2011). Some organizations need security on a particular area and therefore their SCADA will be built with more of the security, focusing on that particular area where the security is needed. Below are some of the SCADA architecture that were examined in order to deduce the generic architecture for this study. However, this study also

recognized that architecture can also be inform of software or hardware architectures. The focus here is in security, therefore the architectures that will be introduced here are based on security which involves both hardware and software.

Fovino, et al (2009) presented in their paper a secure and survivable SCADA architecture. The purpose of the architecture was to be resilient in order to survive cyber-attack. They were of the opinion that the issue with cyber-attack is that the ICS or the SCADA system operates or is connected to the Internet, which the normal security system such as Antivirus and firewalls could not protect due to the magnitude of attacks. However, these systems are built with or are connected with other dedicated systems such as Programmable Logic Controller (PLC) and Remote Terminal Unit (RTU). They communicate using a dedicated communication protocols, such as Distributed Network Protocol version 3 (DNP3), Modbus protocol, ProfiNet protocol and among others. These communication protocols are not secure, which might allow unauthorized access to the system. The dedicated components at which SCADA systems are connected to as described by Fovino et al in their paper as "Production Systems" are not secure. Their architecture made use of Modbus protocol for communication between master and slaves. This is based on the equipment or the production system in use. Modbus is a simple protocol that is not secure. The reason was that, during the time of the creation of the Modbus protocol, security such as integrity, authentication and non-repudiation was not considered. Without security, an intruder doesn't have to do much to penetrate the system. To avoid this, their architecture introduces the following; signature-based communication protocol between master and slaves for authentication purposes. The architecture introduces, time stamp for both ways in order to avoid replay attack and lastly the architecture introduces firewall in form of filtering unit for filtering packets that pass through them. This consists of numbers of filtering units, which makes it difficult for malformed packets not to be detected.

The architecture from Daniel et al (2014) which was built on the 210 MW Thermal Power Plant in Thiruvanathapuram was mainly based on the reliability. Most of the technology that compliments their architecture was to sustain the reliability of the system. They implemented a monitoring system as well as system hot swap and redundancy for reliability purposes. Blanislav et al (2011) presented an architecture that was focused on both security and efficiency. Their architecture was named "oSCADA" which was designed to support huge data points in a distributed environment. The architecture from Dressler (1990) was about reliability and sustainability. He wanted a reliable SCADA architecture that will sustain the growth of the

data in Trans Gas. The system has a backup for the data and failover for the processor and other system components. Novak et al (2013) proposed a SCADA architecture with multi-Agent in a smart distributed environment. Here the Agents are deployed on the PLCs and servers on site and their communication is based on whether they are on the Low-Level Control (LLC) or High-Level Control (HLC). These Agents are for protection of the system for the purpose of reliability and security. The architecture from Coates et al (2010) was about constructing a trust system architecture that can prevent or hinder some unauthorized activities on a SCADA system. This was demonstrated using example of disgruntled employee. However, this demonstrated the extent an insider threat is posed to SCADA system as well as how vulnerable SCADA system could have been without this approach.

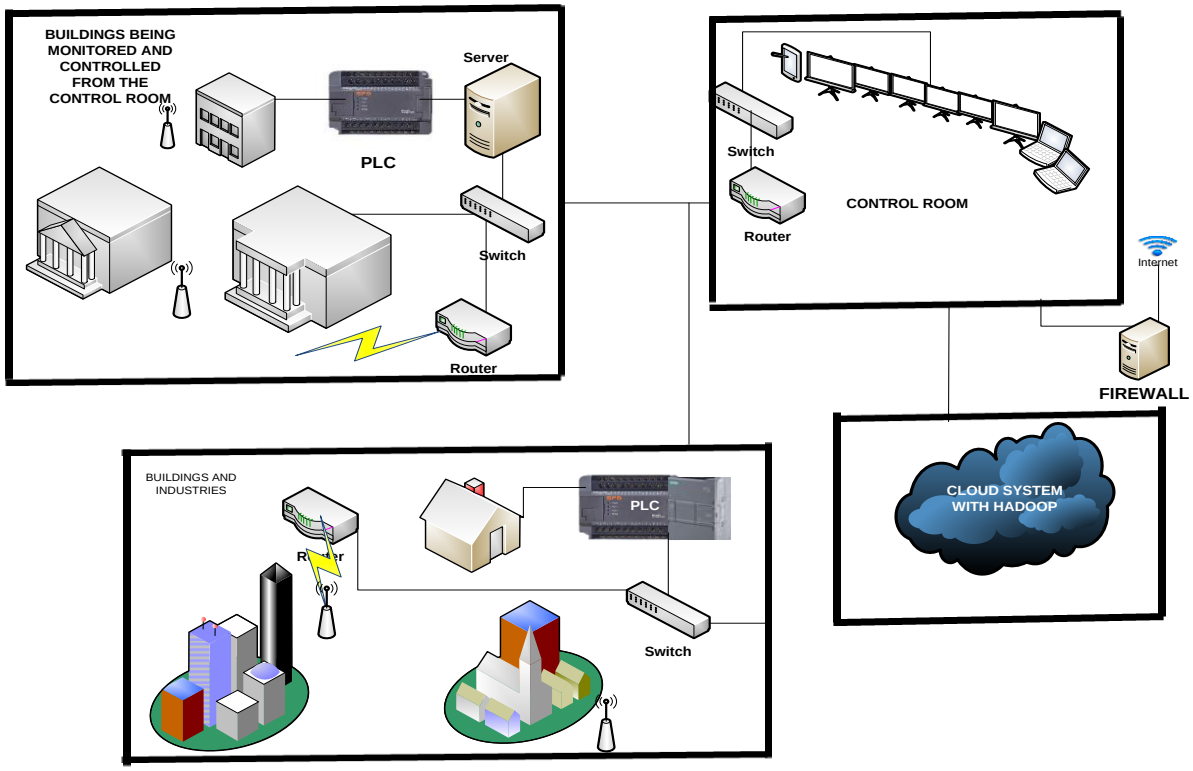


Figure 2-1Common SCADA Architecture with Data Storage in the Cloud and Hadoop

2.2.2 Evolution of SCADA Systems

SCADA system has been in existence for many decades, McClanahan (2002). This has evolved from Monolithic type of SCADA, which was called the first generation of SCADA down to the present-day SCADA that uses Internet of thing. Below are the illustrations of the generations of the SCADA till date.

2.2.2.1 First Generation of SCADA:

The first generation of SCADA was monolithic in its architecture. This early generation of SCADA was developed with only one mainframe connected directly to the RTU. The communication here is purely through proprietary protocols and a standby mainframe was configured for the purpose of failover. The connection between the mainframe and the RTU was made through WAN (Csanyi, 2013). The connection here is between the RTU and the WAN. Hence, other connections are between the mainframe and maybe a redundant mainframe. The reason for the redundant mainframe was in case of any issue with the mainframe, to switch over. The system is mainly a standalone that is not connected to the internet (Ujvarosi, 2016).

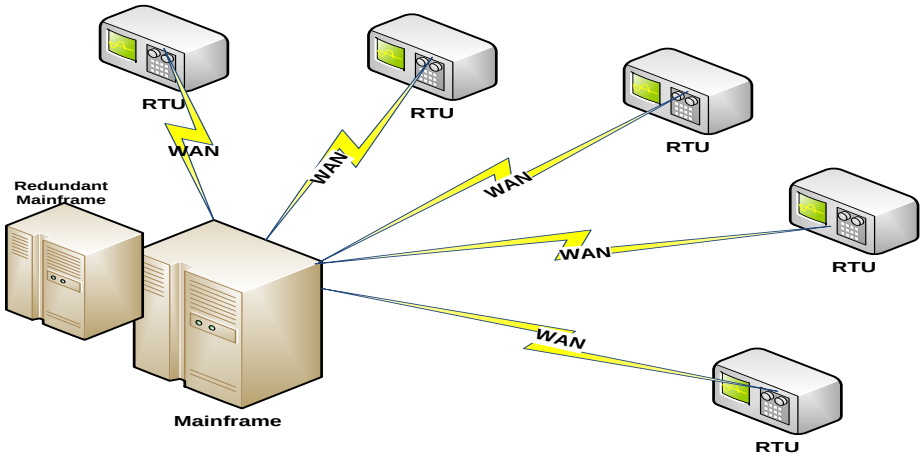


Figure 2-2 First Generation SCADA Architecture

2.2.2.2 Second Generation of SCADA:

The improvement from the first generation was made in second generation by designing the architecture to be distributed. This made use of LAN connection for the local processing across the connected systems. The distributed nature of this generation gave it boost in processing speed. The RTU was connected to the communication station through WAN while the communication stations to the operator station which were connected through LAN. However, the system still uses only the proprietary protocols for communication. The proprietary protocol made it difficult to connect and communicate with other devices that are not from the same vendor. That means as a client, one is tied to a particular vendor for services and every other thing (Csanyi, 2013).

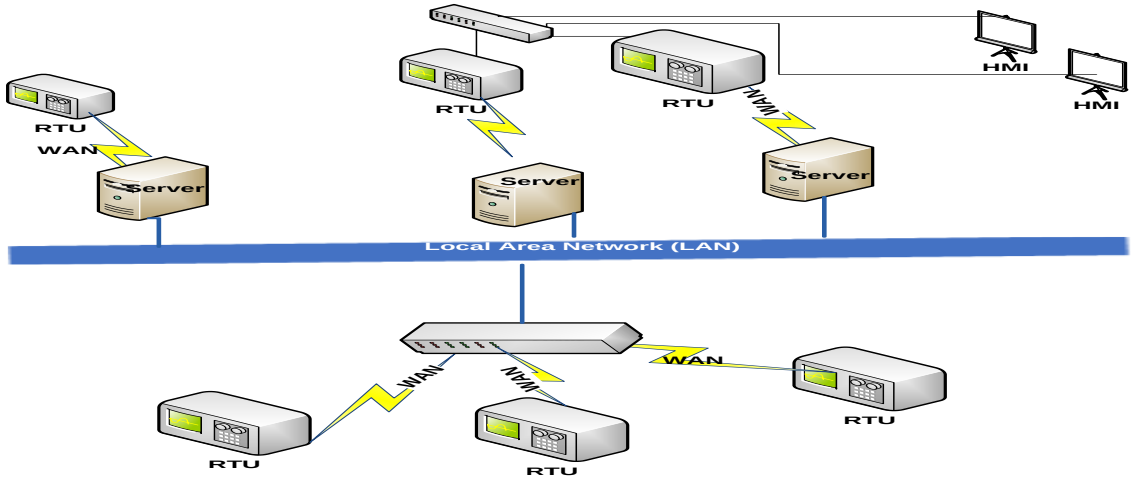


Figure 2-3 Second Generation of SCADA Architecture

2.2.2.3 Third Generation of SCADA:

This generation of SCADA is more closely related to the second generation with difference being that this generation used open standard. The open standard architecture allows the system to use off the shelf hardware and software. This is where the interoperability from vendors kicks off. Devices can communicate with each other even though they are from different vendors. This also made it easier for the system to accommodate more than one LAN network from various locations. This will allow all these systems to be running in parallel making use of one supervisor. Hence, the robustness of the system started here (Csanyi, 2013).

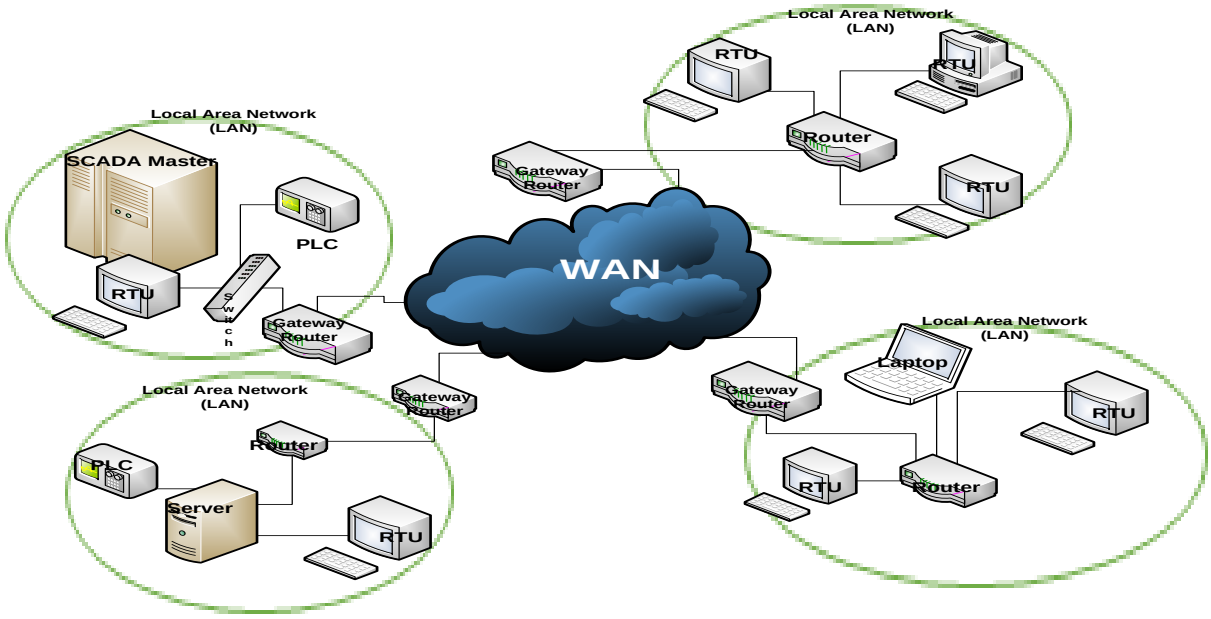


Figure 2-4 Third Generation of SCADA Architecture

2.2.2.4 Fourth Generation of SCADA:

This is the generation that introduces the IoT in the SCADA system as can be seen in today's world. It is now possible to monitor a SCADA system online with your devices. The Internet of a thing coupled with the open standard in the third generation has made it easy to use cloud for the purpose of SCADA data processing and storage. The scalability of the cloud system and open system has allowed off the shelf hardware to be used in the SCADA and lowered the compatibility issues with regards to communication protocols as well as security. This has facilitated the use of Transport Layer Security such as SSL and certificate protocols in order to enhance security of the system (Agarwal, 2014).

Although it was possible to use open network on the fourth generation of SCADA however, security is still issue with SCADA system up till now. The Internet of a thing brought its own security issue, which is currently one of the biggest issues with SCADA security. Most of the recent security holes found with SCADA are due to the fact that the system was connected over the Internet. The protocols were built without security in mind and different vendors have their own design. These have given birth to different kind of attacks and terroristic attacks as compiled by (Internet Security System, 2006), which was presented at the black hat conference. These systems are called critical because of their areas of applications in this present day and time. Hence, the next will look into the areas of application of SCADA system in order to determine the important of the system.

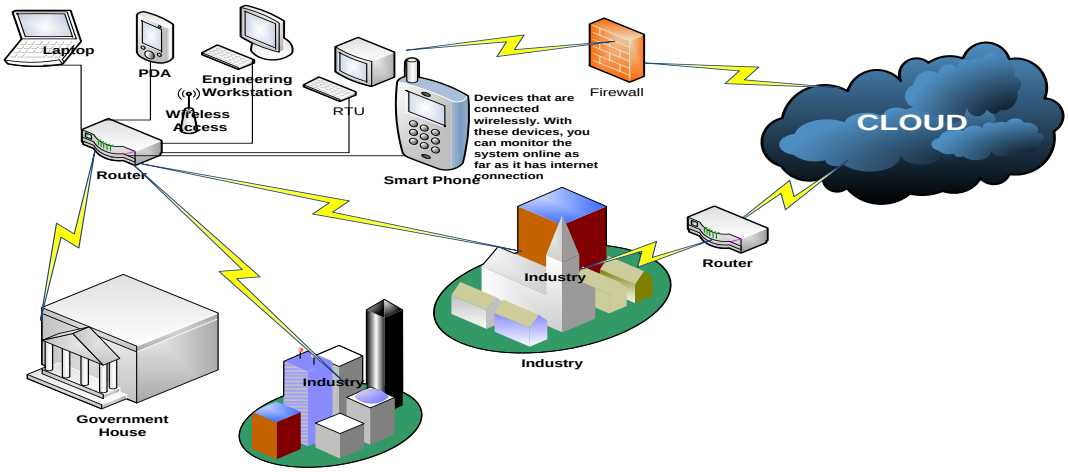


Figure 2-5 Fourth Generation of SCADA Architecture

2.3 Important of SCADA and Areas of Application in Today's World

The advancements in technology in recent time, both in mobile technology, electronics, aeronautics, manufacturing, robotics and among others has posse many questions with regards to their management and operations (ICM, IPI, 2016). The present age of technological revolution has seen increase in the way things are done. The interconnectivity of so many devices and equipment are redefining controls and monitoring. The complexity in managing such an ecosystem in technological environment can be very demanding. The idea of SCADA system for control and monitoring of such an ecosystem ameliorated the difficulties in management of industrial systems. For example, managing and controlling the heating system here at the University of South Wales might be difficult without such control system. Other areas as mentioned earlier are; nuclear system, aerospace management, from locomotives to utility and water managements, from engine management on ships to internet of a thing such as social medias managements will cost more without SCADA (Babovic and Velagic, 2009). SCADA system has lowered the cost and made it possible to control and manage such systems.

As the name implies, Supervisory Control and Data Acquisition, can be seen in most of the systems today. Most of the traffic light in the city are being controlled somewhere as well as the barriers in some parking houses and CCTV. The locomotive uses it in controlling the flow of the system as well as planes that send signal of their location to the control rooms when in the air. The same goes to the military equipment, the power system and other critical infrastructures around (Okeke and Blyth, 2016). Manually controlling and monitoring of these systems can be impossible. Thus, the advancement in technology brought about the advancement in the control and monitoring of the technology. The same way the technology is advancing, threat to this technology is advancing (ICM, IPI, 2016). Many approaches for the protection of these devices have been introduced in the past which are no more effective in the present time. They are obsolete in the sense that they are not meant for these systems, in other word, they might not be useful anymore. For instance, some of the current detection engines might not be able to detect SCADA protocol or multiple stream of flow. This is because the system has evolved from single monolithic to the present-day Internet of Things. Protecting this vital system has taken a new dimension which is like a paradigm shift in the area of IDS. Nowadays in the field of security, some words like Artificial Intelligent (AI), Artificial Life and others have become a buzzword that you hear among security researchers. Below are some introductions to these words;

2.4 Artificial Life

The application of the behaviors seen in the flock of animals is called swarm intelligence (SI) (Beni, and Wang, 1989). Some of the areas of applications of SI so far are in the area mapping and routing, computer security such as IDS. Behaviors of Ants called Ant Colony Optimization (ACO) with regards to food foraging have been applied in IDS (Abadi and Jalali, 2006; Lianying and Fengyu, 2006; Fenet and Hassas, 2001). Bird's behaviors have been applied in other areas such as controlling congestion in autonomous decentralized network (Antoniou et al, 2009) and application of mobbing behaviors in birds as a tool against predators (Davies and Arkin 2012).

Artificial life as an emerging area of research in recent time is a growing area of study. Ackley (2014) has quite made some extensive progress and research on the field of artificial life. His work on the indefinitely scalable computing and artificial life engineering. He produced a prototype called Movable Fixed Object (MFM) and a design of scalable computing. These objects and their behaviors are not deterministic; thus, you cannot determine what their next action will be. They behave in random manner in demonstrating how artificial life should behave. The objects or artificial life showed some characteristics of life by multiplying as of giving birth and some die or were eaten by others.

Artificial life is created by modelling and building it for serving a purpose. The purpose it serves can range from solving problem to being a companion. Since these are artificial, they are built or created with some life like characteristics. These characteristics that are adopted depends on the need and purpose for their creation. Their collective approach to solving these needs or fulfilling the purpose is called swarm intelligence (Rosenberg, 2016).

2.4.1 Swarm Intelligence

Bonabeau et al (1999, Preface) defines Swarm intelligence as "*The emergent collective intelligence of groups of simple agents".*

Garnier, Gautrais, and Theraulaz. (2007) Defines Swarm Intelligence as a community of organised agents with a collective behaviour and decentralized nature of each individual agents. However, both definitions explained the group or community of the agents with collective and organised behaviour. These are common characteristics of swarm intelligence that triggers the study of their behaviour and the integration of it in artificial intelligent. Some of these

behaviours as explained by Bonabeau et al (1999) are division of labour, cooperate clustering and sorting, cooperate nest building as in the case of bees and birds, collective movement of food as in the case of Ants, foraging as a community or group, cooperative way of fighting or distancing themselves from predator and among others. Swarm intelligence is mostly surrounded with the study of some so-called social insects such as Ants, Termites, Bees and among others (Blum and Li, 2008). These animals display some characteristics that made them attractive with regards to problem solving, self-organisation, and division of labour. Ants are well known for its foraging in an environment. Through their pheromone trails, other foragers can locate the food source as well. The same goes to Nest building, their way of division of labour and self-organisation. This study will not go into details of all these characteristics as it is outside the scope. However, it will focus on modelling the behaviour of the swarm for the purpose of this project. The modelling will focus on the movement (self-organised), search or group (foraging), Tasking (division of labour), communication and detection. These are collective behaviours that characterised swarm intelligence compared to multi-agent system where problem can be solved by individual agent (Blum and Li, 2008).

2.4.2 Multi-Agent System (MAS)

Before delving into the field of multi agent system and its application, it would be of great important to define it. The definition of the field of "Multi Agent" will make it easier to understand the subject area. However, the field of discipline of multi agent has been a field with many debates (Kontarinis, 2016). Some researchers believe that it should not be studied independently because of its multidisciplinary nature. Multi-agent System is multidisciplinary in nature because it involves philosophy, economics, mathematics, computer science, ecology and many others.

In order to define multi agent system, the definition of an agent will make it easy to understand. Wooldridge and Jannings (1995; 2012) define agent as a computer system that is capable of functioning autonomously in decision making and interacting with other agents. With the idea of autonomy and interactive characteristics of agent, it is easy to define multi agent system. Based on the definition given by Mike Wooldridge in his book, Wooldridge (2009 P.16). Multi agent system is a collection of agents that can autonomously interact with each other in their environment in order to achieve a certain goal and objectives through cooperation, negotiation and coordination with each other. He believes that agent/multi agent system evolves from the idea of ubiquitous, Interaction, delegation, interconnection and human orientation. Ubiquitous

here means increase in computer processing power and it is everywhere. While the interconnection means that they are connected to each other and interact with each other. Delegation means that more and more human tasks are being delegated to agents to do on our behalf. Figure 2.6 below is the behaviour seen in an agent or agents in the environment.

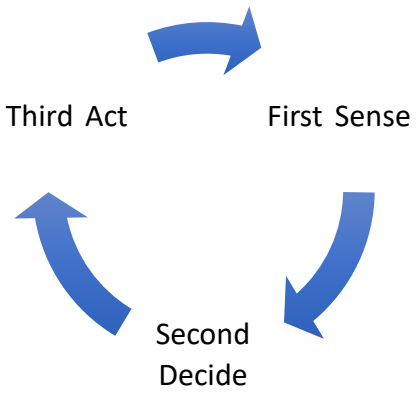


Figure 2-6 Agents behaviours in its environment

Agents in an environment operate as stated in figure 2.6. They are aware of their environments and when a situation is sensed based on their objectives and purpose, they will decide on the best way to go about it. When the decision is made, they will act. The action is the thing that changes the environment. In order to know whether the action was right or wrong, it will be measured against the purpose and objectives. If the actions meet the objectives and purposes, it can be deemed right but when it does not, it can be deemed wrong, Wooldridge (2009 chapter 2).

2.4.2.1 Application of Multi Agent Systems (MAS)

A number of work and research have been carried out with regards to integrating multi-agent systems for solving problems (Kubarkov and Makarov, 2016). As earlier mentioned, the field is growing, and research are being carried out on its application in security as in the case of Wegener (2012). In his published thesis, multi agent was used for malicious behaviour detection in a system. Some agents have been used in detection and control of some system, for optimal operation (Pipattanasomporn et al 2009; Philipis et al 2006; Dimeas and Hatziargyriou, 2006). This shows that MAS can be applied in many industrial control systems for the purpose of security or other usages based on the design objectives.

2.4.2.2 Design of Multi- Agent System

The design and development of multi agent system can be very challenging because of its infancy state at this time (Sycara, 2001; Gasser 2001). Lynch and Rajendran (2004) designed their multi agent system based on agent UML diagram for speech processing in clinical equipment. In their architecture and design different agents have different functions in the environment. One of the ideas behind this project is that agents will be acting autonomously as well as being part of a group. Agents can die as well as exhibits some kind of intelligence in their environment. Pipattanasomporn et al (2009) has similar design, which has different agents for different purposes and objectives; agent for Distributed Electric Resources (DER), control agent, user agent and database agent. Tsang and Kwong (2005) investigated and proposed a similar idea of agent with some definite task. Their work comprises of decision agent, monitor agent, action agent and among others. The agents are organised both in their movement and detection of malicious packet using ant colony algorithm concept. The increase in the activities and operations of malicious intent has brought about new ways of thinking about security.

2.5 Malicious Activities

Malicious definition from Cambridge dictionary is that it is an intention to cause damage to something. It means that the focus is intentional to make the system malfunction or not operational. In the past, systems can malfunction as a result of incompetency or carelessness, which might not be intentional. However, when it is intentional, it becomes malicious and such act is called malicious activity (Cambridge Dictionary, 2017).

Since malicious activities are regarded as intentional activities focused on causing damage or malfunctioning to something. This means that it can as well happen physically or virtually, which is online. Most of the information online on the malicious activities are focused on the virus, worms and trojan horse (Wegener, 2012). These activities are later part of the malicious intent on the computer system. Prior to this, malicious intent has been there. Malicious activities were the reason behind hiding information known as cryptography. Encrypting messages help in protecting it against any malicious intent or from the enemy based on information from SANS institute (2001).

Many scientists and scholars such as Charles Babbage in the 60s down to programmers in the IBM computer and engineers from Bell Telephone laboratories have contributed much to the

success of computer systems. They started out with some ideas of modernising computer and making it accessible by developing some games and theory. However, those ideas were maliciously used to attack computer network in the 70s according to Kaspersky (2017). The first virus on the system was witnessed in the US military computer ARPANET. The program gained entry into the system and wrote on the screen "I'm the Creeper: Catch me if you can". This virus was named creeper virus based on the name displayed on the screen. However, there has been series of other computer viruses and disruptions cause by either virus, worm and trojan. Recently, malicious activities by a computer program can cause physical damage to a system as reported and investigated by (Langner, 2011).

2.5.1 Evolution of Malicious Activities

Malicious activities in this project is with respect to computer system as well as ICS. The network of computers that were created in the past were not created with security in mind because of the knowledge at that time as well as the limited number of connectivity. In the 70s and 80s, there were just a few computers connected on a network. Only few people know and understands the technology behind the system. In the 70s ARPANET has the network of connected computers for their research. This was very exciting that it started attracting interest from organisation, government and researchers (Venkataraman, Brumley and Sen, Spatscheck, 2013). Once people started knowing the technology behind computer, the idea of taking advantage of its vulnerabilities started. The operating system that controls the operation of computers is made up of millions of lines of codes. This cannot be without mistakes and for the fact that something is working does not necessary mean that it does not have vulnerabilities (Perrin, 2010). Over the years as more and more computers are being produced, people started studying the vulnerability on the system which the manufacturers were not even aware of. These vulnerabilities create backdoors to the systems connected together on a network. This can be in form of viruses, malware, trojan horse, and worms that comes inform of program for the computer (Eckstein, 2014).

2.5.2 The Present-Day Malicious Activities

The present-day malicious activities are more sophisticated compared to past twenty years. We are witnessing a technological shift in recent time; in our homes, businesses, schools and industry even the way collaboration is being done. In recent time, the word globalisation has become a household word. People can work anywhere from anywhere in the world and the

issue of location and distance has been lowered. Because of the interconnectivity of systems on the network, the attack vectors are becoming more sophisticated. More and more people have the background knowledge of the operation of computers and industrial network. The result of this knowledges is to look for vulnerabilities or backdoors that was left open by the programmers and penetrate the systems. In today's language, it is called Zero Day Exploits according to Bradley (2016).

Zero Day Exploits are the unknown exploits that have been in the system which is being exploited by hackers. The day the vulnerability is known or made known to the vendor and a patch is provided, then it is no more a Zero Day exploit. This also depends on how many days, weeks or months, the exploits have been known to those that exploits it before it is discovered by the vendor and a solution is provided. The sophistications in the attack vectors in recent time by exploiting this zero-day vulnerabilities to cause damages to systems are obvious (Davies, 2015). According to Bradly (2016), many malicious code writers are using zero-day approach to create virus and worm that will exploit the vulnerabilities in the system. Both the creating of exploits and the exploitation of the vulnerabilities were as a result of knowledge of the technology. The perpetrators are very knowledgeable to recognise vulnerabilities in a system. With this knowledge, systems will be controlled and monitored remotely as seen in the case of Ukrainian power in 2015. The recent attack vectors can unleash damages to physical system such as seen in the case Stuxnet (Langner, 2010). Most of the recent attack vectors are in the form of controlling the system and causing damages to the system. Stuxnet expert langner (2013) when dissecting the exploit lamented that he had never seen this kind of sophisticated worm. According to him, recent attacks will be in this form and this was the first time a worm caused physical damage to a system. The attackers used a certificate that looks like original from a PLC vendor in order to penetrate the Iranian nuclear site and systems. Since the vendor of such equipment might not be aware of the vulnerability, the attacker exploited it to gain access to other vulnerable systems. Another way in which attackers are carrying out their attack is by ransomware programming. This are malware that will lock the system until the ransom is paid as explained by Fruhlinger (2013). Ransomware is a program that is embedded with something that looks like original file. When clicked on the link or file, it will encrypt the system and you might not be able to decrypt it. A lot of big companies have been hit by this kind of attack and these malwares have been reported such as; Wannacry, Petya, Bad Rabbit, Fusob, and among others (Smith, 2016)

2.6 Reasons for Malicious Attack

There are varied opinions on the reason for attacking a system. However, most researchers and industries security experts such as Cloudbric was of the opinion that attackers in these days are different from attackers at the beginning of computer age, Cruz (2015). Attackers in the past, attacks for pleasure and skill test but today it is a different story. In the past, hackers brag about their hacking skills and would like to test their skills on available systems. Some of the reasons for malicious attacks on a system are but not limited to; Information or Cyber Warfare, Espionage, Criminal Activities, Curiosity and many more.

2.6.1 Information or Cyber Warfare

Cyber warfare has become a household word in the mouth of security professional as well as in the news agencies, government and organisations. Some years ago, this was like a science fiction, but today it is a reality that everyone could see. There is an ongoing case in USA with regards to Russia meddling in America election. The information in Hilary Clinton server was leaked to the public and these people believed affected the result of the election according to report from Isikoff and Corn (2017). The Ukrainian power system was attacked, which was reported by Greenberg (2017), Ashley Madison data that was posted online, Iranian nuclear plant that was infected and many others. The Ukrainian power outage as well as the Iranian nuclear plant disruption are some of the cyber warfare as stated by Greenberg. Some hackers whether state sponsored or organisational as well as individuals can either be for the purpose of information such as government secret or individual secrets as seen in Hilary Clinton case.

2.6.2 Espionage Between Enterprises and Nations

Espionage between companies, governments in terms of cyber warfare are on the increase based on everyday TV news and online information (Tucker, 1997). The technological age has seen increase in the way information is being exchanged between organisations and individuals. Majority of people nowadays live online than in the physical world, and this brought about increase in data exchange. Most of the government data as well as organisations are saved somewhere in the cloud (Kenneally, Mulqueen, and Wai, 2014). This can be private or hybrid type of cloud or even public cloud such as Amazon, Microsoft, Google, IBM and many others. In 2009 in Amazon Web Services was discovered a bot net that steal people's password in their server (Giragido and Pirc, 2011, P.15). UK government in a 24 pages report in 2011 announced a strategy to move government services to the cloud. This has made espionage easy between governments and organisations as reported by Nguyen, (2013, P.179).

Most of the reason behind espionage is to uncover the secret of others as well as building trust between allies according to Chung (2014). However, the wake of cloud system has bolstered and made it easy for companies to spy on each other. Everything is connected and through the interconnected devices, companies and governments secrets are stored online. When the operational secret of an organisation is revealed, it is easy to shut down the organisation. The same with the military secret of a country that can give the enemy or criminals upper hand over them.

2.6.3 Criminal Activities

The way criminal activities and organised crime are being carried out now are different from the way it used to be according to United Nation Congress (2015). In the past, criminal activities use to be physical things. Things are taken away physically but now, everything is organised online. Even people who have never met themselves face to face do organise some criminal activities online. The group called anonymous are there online hacking into computer systems. London riot in 2011 was planned and organised on Facebook and Twitter as reported on the Guardian by Halliday (2012). Malicious activities just to name a few starting with online extortion, Distributed Denial of Service (DDoS), unauthorised access to computer system, Pharming which is online fraud activities, phishing, spam, spoofing and others are all criminal activities. Anyone found engaged in any of these activities can be sentenced to many years in prison.

2.6.4 Curiosity and Fun

In the past a lot of malicious activities were done out of curiosity and fun. Such malicious activities ranging from defacement of website, web content change, Denial of service and many more as explained by Carafano, (2011, P.112). These were done in the past for the fun of making others look the way they want it. Sometimes hackers hack because they would like to know the content of the system or what you are doing. This is where curiosity plays a very big role in the way people hack and their reason for hacking. But all these in the past were not intended to cause damages to the system. Just as Xu, Hu, and Zhang (2013) stated that when this approach of curiosity is left uncontrolled, it will turn into criminal activities. They were of the opinion that some talented individuals in the area of computing are turning into hackers if no moral values are present. These group or individuals if not controlled early in life will get in contact with criminal gangs on the internet or elsewhere and receive more induction and

training. This induction can start from school, home and companions by their peers and colleagues as well as their personal search.

2.7 Malicious Activities Detection Systems

In the era of technological connectivity as we are seeing now, malicious activities are bound to be present as some individual would like to abuse the system as stated by Almaatouq et al (2014). In the past, the misuse of the system was for fun and curiosity. However, now the idea has moved from curiosity and fun to malicious intent such as causing damage or extorting money from the owners. As explained earlier on some of the reasons for malicious intent such as cyber warfare, espionage and among others. These behaviours can either cripple the system or causes the system to malfunction if not detected on time. Because of the intention behind this kind of behaviours, a lot of measures have been proposed and implemented in detecting such behaviours.

According to SANS institute (2001) there are two types of Intrusion Detection System (IDS); host based, and network-based IDS. Host based IDS operates from the host machine to protect the machine from any entry to the system. They operate by collecting information and monitoring a single system. This kind of IDS depends on the information from the audit trails of the system as well as the logs. The information collected will reveal the functionality of the single computer as well as the network connected areas. However, Network Intrusion Detection system (NIDS) operates on the network. This kind of IDS is placed on the network to monitor the whole network system. Information is collected on the packets that flows in and out of the network. These packets are monitored for anomaly based on the prescription and configuration of the NIDS.

Summary

This chapter dealt with the background of some important concept in this project. It is benefit both those that are familiar with security and those that not familiar with some of these security terms. Those that are familiar with security will refresh knowledge from this. The chapter explores some background knowledge of the Industrial control system as well as some part of it such as SCADA system. This exploration was filtered down to the evolution of SCADA system from the first generation to the fourth generation of SCADA. The fourth generation of SCADA are the current state of the art SCADA systems that are connected to the internet. This is the SCADA system that deals with the Internet of Things (IoT). Some of the SCADA equipment were introduced such as PLC, HMI, sensors and other equipment. Their communication protocols were introduced such as Modbus, ProfiNet and DNP3 as the most used protocols in Europe and north America. The areas of SCADA systems application were explained and the concept of artificial life, multiagent system and swarm intelligent for the protection of computer system were explored.

This chapter highlighted the background reasons behind malicious activities in this present time. As the technological sophistication is increasing the same way the attack vectors are increasing. The previous attack vectors and the present were compared, and this gave birth to the future attack vectors or the prediction of what the future will look like. This explains what we are to expect and how to prepare for it. The attacks in the past were mainly for curiosity and the fun of it but the present-day attacks are mainly for malicious intent. The landscape of attacks is changing, the same way the detection of these attacks is changing also.

In summary, the project explored the present challenges and hypothesised a possible solution in solving those challenges. The challenges the present SCADA system is facing with regards to securing it is enormous. The current attacks vector is causing physical damage to the system as seen in the case of Stuxnet and Ukrainian power station. The question is how to reduce such attacks from having a devastating effect on the system? The hypothesis has been stated as well as the aim and objectives of the study. This chapter have explored in detail some of the concept that will help in understanding the project. The next chapter is to delve into literatures and to start answering those questions.

CHAPTER III

3 LITERATURE REVIEW

The previous chapter was on the background information of the project and some basic technological term that are associated with this project. These are some words that will appear in the course of this study as it progresses from chapter to chapter. This is to help in familiarising oneself with some possible nomenclature as well as current words in use here. Understanding this will aid in following the study as it progresses along with the chapters.

This chapter will deal with the review of both past and current literatures in the areas of security, computing, ICS and SCADA systems as well as their relevant to this project. This chapter will take the background information further and dig deeper into the subject area in order to find connection with the subject area of this project. This will start by exploring further into the types of anomalies and their representations on ICS as a whole and SCADA system in particular. The available approaches for the detection of these anomalies will be discussed such as the current available IDS, the techniques that are built into these IDS that made them capable for the task. The present detection approaches will be examined, and this will give an insight into the future detection approach. Some of the approaches that are related to the proposed approach will be examined and some distinctions will be made between them.

The proposed approach is using artificial life in securing Industrial Control System. The proposed artificial life in this project is the flocks of bird approach in detecting predator. This chapter will review some of the important characteristics seen in birds that distinguishes them from other animals. How do they detect predators when they are in flock? Some of the existing hypotheses that are related to the flocks of bird will be highlighted and cited in order to support the idea. This animal has some characteristics that worth emulating such as their detection approach. They can flock in thousands and even millions but targeting one among them can be very difficult. Each one of them in the flock can detect predator and continue to flock with the same flock. Their way of communicating danger to the rest of the flock is phenomenal and worth studying. These are some of the characteristics that made them perfect to be emulated for the protection of Industrial Control Systems.

3.1 Industrial Control System Anomalies

Anomaly here is when things deviate from normal or so-called prescribed way. This can be attributed to so many things such as behaviour, operation, access and many others. Anomalies have been found on almost every part of Industrial Control Systems ranging from PLC, HMI to sensors and even the protocols to engineering workstation. Industrial control systems have witnessed so many anomalies in the past few years and the common one as described by Morris, Vaughn and Dandass (2012) are Denial of Service (DoS), response injection, command injection and reconnaissance.

The attack on the Iranian SCADA system was one of the wakeups call to protect the process control system. The virus was labeled Stuxnet, which penetrated the system through the vendors Driver DLL. The Driver DLL was infected which gave the attacker the privilege of penetrating the control system and multiplying, infecting and causing damages to the system (langner, 2011). Whether this is called a malware, or any other anomaly behavior tools, their behaviors in the system cannot be normal therefore, they will cause the system to malfunction. Malware can be in form of virus, Trojan, Worm, Botnet and among others that has destructive possibilities or malicious intent.

Over the years, there have been reports on the increasing numbers of malicious activities on computer systems. In a report from Godin (2011), a patient insulin pump was hacked, and the dosage was manipulated. This made the system to malfunction by coursing the software that control it to malfunction. Malfunction software is the reason for malware. The idea of malware on a system can be dated as far back as 1971 with the Creeper virus. The virus copied itself from an infected system to other systems on the network (Chen and Robert, 2005). There have been numerous reports of similar attacks on various systems both personal and industrial systems such as “Festering Hate", "Cyberalds,", "Elk Cloner” and among others (Spafford, 1990). One thing to deduce from the characteristics of all the malicious behavior as reported here is that, they all performed unauthorized activity on the infected system causing the abnormal behavior of the system. Their behaviors were not in line with the predefined programs on the system they attached themselves to. Below are some known types of anomalies that can be seen on ICS and their behaviors.

3.1.1 Contextual Anomalies

Contextual or conditional anomalies are the type of anomalies found in a specific context as stated by Songs et al (2007). This type of anomaly is found in the structure of the data based on context and behaviour of the data. This can also be a system based on the context at which the system is defined and the behaviour of the system. Therefore, there should be a determinant based on the context of the data available. The location of a bird at a particular point in time can be determined based on the information available such as their speed, time and direction. Behavioural in other word is not contextual in the sense that it is based on a particular behaviour on a certain place or things to determine in order to make an informed decision. The collection of the behavioural characteristics can give the overall behaviour of the system. in determining contextual anomalies on a system, behaviour of the system within a certain context is very important. There are some certain behaviours that are normal on a place while on another place it is not normal. This kind of behaviours is an anomaly for the location that deemed it as abnormal and should be detected.

3.1.2 Point Anomalies

This type of anomaly is one of the simplest types of anomaly that deals with the abnormal data in the mist of other data (Hays, and Capretz, 2015). This can be something that fall outside the normal state such as “moving” in programming a train system to be moving at the speed of 100 miles per hour and suddenly, the train is moving faster or slower that the normal speed. This condition has fallen outside the set point and will be detected as an anomaly. This can cause the system to slow down as well as collide with other train if not detected on time. If it is moving faster, it might also collide with other train from behind and if it is moving slower, other trains will collide with it or vise vasa. This can be applied to anything for the detection of the anomalies in the system. Songs et al (2007) and some other researchers have used example of spending that goes out of someone account. This is for the detection of fraud in the system, therefore the account has been set for a limit. When the spending on a certain day from a location outside the city is above the threshold, this is an anomaly in the system based on the spending alone.

3.1.3 Collective Anomalies

Zheng, Zhang and Yu (2015) define collective anomaly as the collection of anomalous datasets from different locations in different times. This is to say that these datasets will be used to determine anomalous behaviour in a system. A single dataset might not be enough or dataset

from a single location might not solve this particular issue as the data must be compared with the rest of the data. Individual data might not be anomalous but the collection or the mixing of it with other datasets that are anomalous or from other location can make it anomalous. Songs et al (2007) gave an example with the data collected in computer such as ftp, buffer overflow, ssh and among others. This information together can be anomalous on a system but the individual data such as ssh might not be. Therefore, collective anomaly deals with the collection of different data or information that can prove anomalous from different locations against the collected data. The same way the locations of individual trains can determine whether there is going to be a collision in a place or not. A train in a location is not anomalous but two or three trains on the same track, driving both clockwise and anticlockwise direction is surely an anomaly on the system. The reason is that, the train will surely collide with each other if no measure is put in place for the detection of such anomaly.

3.2 Anomaly Attacks on Industrial Control System

Since the emergence of Stuxnet virus on the ICS, there have been numerous reports of attacks on ICS in many countries. Singer and Friedman (2014) spoke of the danger ahead with the ICS as a result of the connectivity. The danger can be seen from the current discoveries of the vulnerabilities on the ICS that happened from 2010 till date. The increase is phenomenal based on the survey from Dell and other organisations. The information in figure 3.1 showed that the most common attack vector is buffer overflow which amounts to almost 26% of the total attack witnessed in the year 2014 and some part of 2015 according to Dell. The industry that witnessed most of the attack are the manufacturing industry as seen in figure 3.2. The evidence of the growing attack on the ICS can be seen from the data in figure 3.2 from 2012 to 2014. In 2014, the amount of attacks witnessed in Finland alone is more than the one recorded in 2013 for the whole world. These were only the reported attacks because the number is far higher than what was recorded.

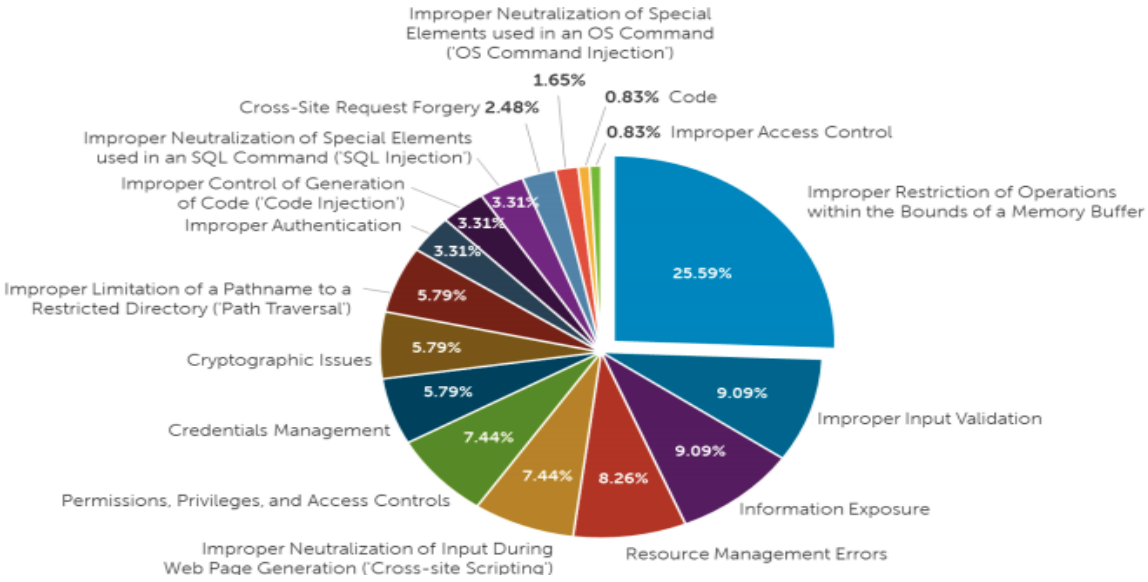


Figure 3-1 SCADA Attack Methods. Source Dell (2015)

Year	Number of Attacks
2012	91, 676
2013	163,228
2014	675,186

Year 2014	Number of Attacks
UK	69,656
Finland	202, 322
USA	51,258

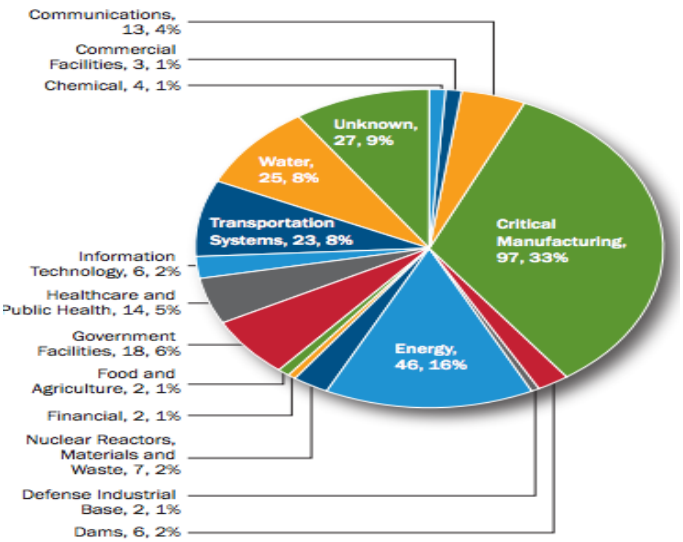


Figure 3-2 Most Affected Industries, Source Dell (2015)

The problem with the attack of a system is that most organisation are not even aware that their system have been attacked. The same with the Iranian plant that was attacked since 2009 and only came to recognise it in 2010 as stated by Langner (2011). The report from SANS Institute (2017) stated that 40% of the organisation do not know whether their system has been infected. In 2015, the global attack based on the analysis of the internet connectivity surged up to 8.19 billion (Rock, 2016). Their analysis revealed that there was 54 percent increase each month in 2015 compared to 2014. Dell reported also that in 2015, they blocked 2.17 trillion attacks which is about 73 percent increase compared to 2014 and identified 64 million unique malwares in 2015. If the number of malwares are recorded in millions, the attack might be greater in the subsequent years. Kaspersky (2017) recorded the effect Wannacry ransomware had on over

200, 000 systems in 150 countries of the world. If one malware like Wannacry could have such magnitude of effect, what about 64 million unique recorded in 2015?

The reports from some of the security companies such as Dell, Kaspersky and others, suggested increase in the amount of attack to ICS. The danger is very high due to ignorant of some organisations with regards to security. The connectivity of these devices on the internet has created another danger for the systems. The problem is that, most of the systems are still using some of the outdated Operating Systems such as Windows XP and Windows 2000 (Dell, 2015). These operating system does not have security patches anymore because their vendor does not patch them anymore. Romania car plant as well as UK Nissan plant were affected by Wannacry and resulted in days of no production. These were days of loss of production and money.

3.3 State of the Art Detection Approaches

Intrusion detection systems can be categorised into signature based such as using Snort rules, and anomaly detection by detecting abnormal behaviours (Bi, Zhang, Cheng, 2009; Anderson, 1980; Denning, 1987). In the past, detection is mainly carried out using manual analysis of the logs by the IT personnel by analysing the log data for the purpose of detecting anomalies. However, this shows that the concept has been around for many years, but something changed in 1980. James Anderson presented a paper titled Computer Security Threat Monitoring and Surveillance (CSTMS), Anderson (1980) which showed that audit trails contain information for detection of misuse on a host-based computer system. The idea from Anderson gave birth to the first IDS or named Intrusion Detection Expert System (IDES) by Dorothy Denning in 1983. The IDES was based on the analysis of audit trails of the mainframe computer from the government. This gave birth to other research on using the audit trail and known misuse behaviour to detect misuse on a system (Denning, 1987).

The present-day commercial host and network IDS was first introduced in the 90s for monitoring and gathering information both on the network and host computer. The work was initially instigated by haystack Labs that introduced Distributed Intrusion Detection System (DIDS). This was extended to other IDSs such as using some algorithms and programming capability in detecting anomalies and misuse on the system as well as using some rules such as Snort rules for detection. Some of these rules are built into firewall and antivirus on a network or host in order to enhance control measures on the system. Internet Security System (ISS) as one of the big players in IDS, developed by RealSecure in 1997 as the first commercial network

IDS for windows NT 4.0 (Bruneau, 2001). These tools are signature based, which involves matching the signatures against the changes in the system. However, these are based on the known attack and cannot detect unknown attack.

The term signature-based IDS has been there for many years and have been applied in many systems for the detection of intrusion. Signature based as put together by Brox, (2002) is pattern recognition approach based on the behaviours of attack stored in the database. These behaviours that the attacker exhibits are named signature when known and studied. The signature-based IDS is to recognise these patterns, based on the instruction on the behaviours stored on the database. These patterns are known and studied, which is one of the weakness of signature-based IDS, they only operate on the known attacks. Signatures are in form of rules such as Snort rules that is popular with signature-based IDS.

Signature based IDS has been used in many computer systems for the protection of the system. Patron, Yurcik and Doss (2001) showed that signature-based IDS can be attacked by disabling the alarm and penetrating the system without being noticed. This come by crafting a packet in such a way that it seems to be a false positive alarm. This vulnerability was named "Squealing" in their paper. Anjum, Subhadrabandhu and Sarkar (2003) in their Knowledge Based Intrusion Detection (KBID) of known attacks was carried out. This was a signature-based detection approach of the attack on the ad hoc routing protocols. Their consideration of detecting the intruder based on their connections between the source and destination nodes. However, the conclusion was that a proactive routing protocol is more effective compared to reactive counterpart. The problem of mobility in detecting intruder in an ad hoc network was considered especially for reactive routing protocols. Verba and Milvich (2008) incorporated both signature-based, and anomaly-based IDS for the detection of attack on SCADA system. The attack model that was analysed here are fuzzing and Man-In-The- Middle (MITM) attacks. This showed that signature-based detection approach has been used for ICS, although considered now obsolete. Hence, the idea must be acknowledged and maybe there is a way of using those hybrid approach again.

The new evolution of IDSs are being developed which are based on using artificial intelligence algorithms such as Bayesian Neural Network (BNN) and Artificial Neural Network (ANN), Swarm Intelligence algorithm such as Ant Colony Optimization (ACO), Decision Tree Induction (DTI), Support Vector Machine (SVM), Next-Generation Firewall, Bro-IDS and

many more. However, some of these present IDS have some bottleneck to overcome such as memory usage, Big Data issue, which is the amount of data network or system generates such as volume, velocity and variety. The amount of data networks generate can be overwhelming for some IDS in terms of processing and storage, which the current IDSs are facing as one of the challenges.

3.3.1 Future Intrusion Detection Approaches

The future of IDS will be based on the current challenges of the IDS, which are Big Data challenges such as sensor data and Internet of Thing (IoT) or unstructured data. According to report from IBM in 2012, almost 2.5 quintillion data were created every day and it's increasing (IBM, 2012). Most of this data are both unstructured and structured, such as data from SCADA sensors and social networks. In order to address these issues for the future IDS, Elngar et al (2013) proposed the use of feature selection in reducing the amount of data. However, Jeong et al (2012); Suthaharan (2013) proposed and demonstrated the use of Hadoop and its features in minimizing some of the Big Data problems such as storage and data processing. Some of these experiments showed the viability of using Hadoop framework in resolving some of the current IDS challenges with regards to distributed data processing and storage. As stated above, some of the current IDS research are focused on the application of biological means in solving some of these big data challenges by studying how these social insects and animals solves some difficult problem. The methodology has shown a massive improvement over the years as more researchers are deeply engaged in it such as Ant Colony Clustering Model (ACCM) in a distributed environment or distributed processing (Tsang and kwong, 2005). Fenet and Hassas (2001) first demonstrated the use of Ant Colony Optimization (ACO) for IDS as distributed IDS and many others. However, the idea has been a promising area for IDS in minimizing the challenges of Big Data and others as stated above.

Due to the technological advancements, the speed at which most of the technologies are evolving, ways of securing these technologies are changing as well. Manual processing of computer logs was possible in the past due to the technology at that time. However, manual processing of computing logs in big organization is deemed impossible due to the volume of data. In the past, data are in bytes and kilobytes while currently data are in gigabytes, terabytes and petabytes. Some of these data are unstructured and from different sources and needed to be correlated and structured, analysed and stored. The traditional IDS cannot handle this kind of data both in volume as per the amount of data, the velocity as per the speed at which these

data are processed and the variety which refers to the data with different structures and from different sources. As the idea of integrating IDS in the cloud or using cloud in minimizing these challenges are evolving with promising future as seen with the application of ACCM, ACO and among others. With a well application of the concept and good architecture, the issues as seen above can be minimized.

3.4 Intrusion Detection Techniques for Anomalies Detection

The survey from some of the intrusion detection in section 3.3 outlined the present and future detection approaches. Things or systems can be characterised as abnormal when it falls outside the normal condition. Some conditions are normal based on the prescriptions and some can be characterised as abnormal. ICS is very important in any area of application because lives may depend on it. Many industries nowadays operate ICS such as SCADA systems for easy operation of their industry. A report from Germany in 2014 was about the development of ambulance from a German engineering graduate Alec Momont. His drone ambulance prototype as reported by Prigg (2014) is capable of administering first aid assistance through some remote command and control. This was also evident in the recent development in the insulin pump that uses wireless connection. The pump comes with the possibility of adjusting the level of insulin remotely, which poses a lot of security issues (Maisel and Kohno, 2010). All these recent technologies need protection and security for normal operation. For example, the insulin pump was once attacked, and an overdose was administered to the patient. These are anomalies in the system which can be caused by either the outsider or the insider and below are some of the detection techniques that have been used in the protection of ICS.

3.4.1 Ant Colony Optimization Detection Technique

Behaviors of Ants called Ant Colony Optimization (ACO) with regards to food foraging have been applied in IDS (Abadi and Jalali, 2006; Lianying and Fengyu, 2006; Fenet and Hassas, 2001). Ants are known for their collective behavior in search of foods and such approach has been followed and emulated for securing ICS. This animal locates food sources based on the pheromone trail from others. When an Ant locate food, or on the process of food search, it leaves pheromone on the path so that others will follow the same trail in locating the food source. This approach has been adopted in many researches that gave birth to solving some problems such as travel salesman problem, routing systems and intrusion detection systems.

Tsang and Kwong (2005) proposed the use of Ant Colony Clustering Model for detecting malicious activities on the ICS. Their approach was using Ant colony algorithm for searching, detection and prevention of malicious activity on an industrial control system. This approach was named Ant Colony Clustering Model (ACCM) which was made up of some categories of agents namely; Monitor, Decision and Action, Coordination, User Interface and Registration agents. These agents are placed on the server or the PLC/RTU for any incoming packets. The monitor agents will monitor the traffic but could not decide whether a packet is malicious or not. The information gathered from monitoring will be passed on to decision agent that will place the packets in cluster and passed it on to action agent. The action agent will take action based on the observation or the parameters that was set. The coordination agent will now work on the clusters from the decision agent while the user interface agent will be responsible for the communication between the user and the control station by translating their request and displaying also to the user the condition of the system. The registration agent will control all the agents' functions in order to facilitate their communication. This approach was tested using dataset from KDD-Cup99, ACCM effectiveness in detection of malicious behaviour was compared against K-Mean and EM algorithm and their result showed that ACCM was more effective in detecting malicious activities compared to other two.

3.4.2 Machine Learning

The term Machine Learning (ML) was coined by Arthur Samuel in the year 1959 according to Kohavi and Provost (1998). ML is a field of science that deals with studying and learning the behaviours and pattern of data in a system for filtering the unwanted as well as for the detection of anomalies. This algorithm has been used for the protection of computer systems as well as email filtering, data mining, character recognition as well as object recognition and among others. Machine Learning algorithm can perform both supervised and unsupervised learning approaches. These are approaches where either the training and testing data is given, or it is not given, and the algorithm will learn from the pattern. This algorithm has been applied in detection of anomalies on ICS.

below are some descriptions of some ML algorithm such as ANN, KNN, SVM and AIS. These are not limited to the ones explained above, there are others that are closely related to the ones explained above. Yasakethu and Jiang (2013) in their paper titled "intrusion detection via ML for SCADA system protection explained some other machine learning" approaches such as Hidden Markov Model (HMM), Rule-based Approach and One Class SVM. Their paper

investigated these approaches and cited some of the references where the approaches have been applied on SCADA systems. ML is a vital algorithm in artificial intelligent that has been applied in many systems.

3.4.2.1 Artificial Neural Networks (ANN)

Artificial neural network acquires the name from the biological neural network or network of neurons in the body. This is a collection of interconnected nodes or units that processes information in a parallel manner (Schalkoff, 1997). This is an abstraction of biological neurons in the body with millions of connections. This phenomenon is very complex in the way they are connected to each other and the way information is passed to each other. The same way, the emulation of it in solving complex computing and network issue will be advantageous.

The ANN can learn the operational configurations of a normal system in a network as explained by Schalkoff, (1997). The is the same way the brain learn from the environment based on the experience. The ANN can generalise the learnt values of the normal to others. This means that they can distinguish abnormal based on the learned normal and anything outside the learnt normal is abnormal. This approach has been applied in IDS for securing computer systems such as in the case of Wei and Hao-yu (2010). The IDS is based on Back Propagation Neural Network (BPNN). There are other types of IDS based on Neural Network such as Self Organising Map (SOM), Support Vector Machine (SVM) and Simulated Annealing Neural Network (SANN). These are all types of ANN that can either be applied alone or use in combination with others for a better detection purpose.

The BPNN was actually developed by Rumehart, Hinton and Williams (1986) in order to solve the problem of feedforward ANN training. After each training the error back propagation is used to adjust the network. The nodes are equipped with a weight that will be adjusted in the process of training in order to produce learning. SVM on the other hand is the type of approach that deals with data classification, regression and for the detection of outliers (Gunn, 1998). Its either the data belongs to one or the other group based on the classification of the group members. SANN is a probabilistic approach for detecting global optimum. It approaches problems in a heuristic manner by generating solutions for optimisation issues. The probability there is the action the algorithm performs in order to determine the best state. This can be between the old state and the new state of the system. This has been applied in travel sales

problem for determining the best route to take. The shortest path in some way might not be the best path (Boese and Kahng, 1993).

3.4.2.2 Artificial Neural Network Based IDS for ICS

ANN IDS has been applied in many computer systems for detecting anomalies in the system. Ching and Huang (2010) used ANN IDS for the detection of intrusion in form of stepping stone. These are the type of intrusion where the perpetrator tries to mask himself behind another system to avoid detection. The perpetrators go through other computer to the destination computer. The detection approach of such intrusion was based on calculating the Round Trip-delay Time (RTT). This is by calculating the time the message was first released and how long it took it to get to the destination. This detection is mainly to estimate the downstream length. The estimation will be from the system that is being used to detect the attacks direct to the victim's system. The likelihood of an attack in a system is detected based on the length of the downstream.

There have been several occasions where ANN has been applied in combination with other detection approaches for improving their detection mechanisms. Research from Mafra, Moll, and Fraga (2010) introduced the Intelligent Intrusion Detection System (IIDS). This was a two layers approach and the first layer was implemented with Kohonen Neural Network (KNN). The KNN helped in reducing false negative rate while the second layer was implemented with SVM. The SVM was used to improve detection of anomalies in the system. However, the focus here is the application of this IDS on ICS for the detection of anomalies. Linda, Vollmer, and Manic, (2009) implemented neural network intrusion detection system for securing critical infrastructures. This was a semi supervised IDS that was modelled with Neural Network Model (NNM) with data from the power station. Their model applied Levenberg Marquardt and error back propagation as well as features extraction from windows OS. The features that were extracted and checked were IP addresses, packets interval, zero window size packets, protocols and many more. The data that was recorded from the devices were around 20 000 packets between the workstation and the PLCs. This show that the application of ANN for securing ICS is possible and viable but is it suitable for today's bigdata?

3.4.2.3 Artificial Immune System

Artificial Immune System is another type of algorithm that is based on the function of the immune or the emulation of the immune system in vertebrate. This is a type of machine learning

based algorithm that is focused on learning, pattern recognition as well as memory. The idea is dated back to 80s with the model from Farmer, Norman and Perelson (1986) and earlier work of Jerne (1973) and Jerne (1974) on immune system and network theory of the immune. The same way the immune system protects the body by fighting the abnormalities that try entering the body, the artificial immune system is meant to work that way also. In other way, it is meant to emulate the way the body immune system works. This has been emulated and applied in computer systems for the protection of the system as well as the ICS.

Bere and Muyingi (2015) in their paper accessed the use of Artificial Immune System (AIS) for securing ICS. Their assessment of the viability of its application on ICS was that it is possible and a viable option. They examined the operation of immune in the body and how it protects the body from foreign body or attack. The same approach can be adopted in protecting ICS with the full understanding of its operation. The challenges as they mentioned was the understanding of the concept as many people do not understand it yet. Ehret, and Ultes-Nitsche, (2009) went further in designing and implementing an IDS based on AIS that can detect both misuse and anomaly on a system. The same way when body is attacked with new sickness, it slows down the operations and maybe weakens the person, the same way system can be slowed down to alert the operators. The signature can be recorded for the anomaly detection. This idea was implemented in ADENOIDS. This is an indication that AIS can be used for securing ICS, but the question here is in the detection of distributed attacks, can the system detect multiple attacks at the same time? The answer to this question has not been found because the idea of AIS is detection and fighting one thing at a time.

3.4.3 K-Means

K-Means is the classification method use in the separation of the closely related data. The algorithm is based on clustering data that are closely related based on observation and the idea can be dated back to 50s. Some of the ideas are named Lloyd Forgy because of his popular publication Forgy (1965) on Cluster Analysis of Multivariant Data which is focused on measuring Efficiency Versus Interpretability of Classifications. However, the approach has been there as early as 1956 by Hugo Steinhaus who is a mathematician and proposed this approach with view of multidimensional study (Steinhaus, 1956). Today the approach has been adopted in IDS for securing computer system and most importantly for this study, the ICS security.

Eslamnezhad and Varjani (2014) proposed the use of MinMax K-means clustering to improve clustering experience of data. The application of the method was based on the cluster weighting and the variance, the cluster with higher weight will be receive by the cluster with bigger variance. However, the approach solved the shortcoming of the simple K-means in the area of initial centre and to reduced false positive. Kiss, Genge, Haller and Sebestyen (2014) proposed a statistical method based on K-means clustering for detecting anomalies on ICS. They investigated different anomalies found and were able to group them based on their characteristics. Hadoop framework MapReduce was used for the processing of data which solve the problem of bigdata and time. This and many other approaches that is based on the application of K-means for anomaly detection shows that it is a viable option for detection of anomalies on ICS.

3.5 Artificial Life (AL) for Anomalies Detection on ICS

Artificial Life in chapter 1 and 2 was define as an emergent property that is created with and for a purpose that exhibits certain pattern of life. One of the things to notice here is the definition of agents or multiagent system. sometimes the two words are used interchangeable to mean the same thing. Wooldridge and Jannings (1995) define agent as a computer system that is capable of functioning on its own and make autonomous decision. These are patterns of their existence as agents, meaning that a computer system that is capable of functioning autonomously can be regarded as agents. This is why there is confusion in the definition of the two concepts; agents and artificial life. However, on this note, the focus here is the AL and the two words will be used interchangeably to mean AL.

Above from section 3.4 are some of the AL that have been used both as IDS and in securing ICS. Tsang and Kwong (2005) as explained in section above used Ant Colony Optimisation in securing ICS. This was applied as intrusion detection system for securing ICS and these agents have different functions and collaborate with one another for the detection of anomalies. Shosha et al (2011) proposed an IDS architecture called Distributed Intrusion Detection System (DIDS) for securing SCADA systems. The proposed architecture involved correlation of information to enable the detection of anomalies. Information is gathered from the outstation and from the control station in order to help in detection of correlated attacks. This information is used as an intelligence, which allows an agent in making an informed decision on the issue of vulnerability. General Network Protocol Traffic (GNPT) and the Specific SCADA Protocol Traffic (SSPT) are the two-proposed detection technique used. The GNPT detects general

network attack by observing the network behaviour or some reconnaissance while the SSPT detects attack or anomaly that comes through SCADA protocols. The Local Detection Agent (LDA) is assigned to the outstation while the Global Detection Agent (GDA) is assigned to the control centre and both will correlate information that will help in detecting anomaly behaviour on ICS. This is similar to the coloration and information transfer seen in the flock of birds.

3.6 Collective Behaviours in Birds

The study of Birds behaviours belongs to evolutionary computation technique called Particle Swarm Optimisation (PSO) that was developed by Kennedy and Eberhart (1995). The study mainly focuses on the attributes exhibits by these social animals that can be used in computational models in solving complex problems. Each bird in the environment is named particle for the purpose of particle theory and for better understanding. The mathematical equations for explaining three important behaviours that will be useful for this project such as their convergence, separation and detection as explained by (Shi and Eberhart, 1998);

K = The convergence factor

φ = The sum of the individual and group learning factors as explained above which is $\varphi = c_{1+}\, c_2 > 4$

V_{id} = This is the velocity of the *i*th particle in the *d*th dimension. The *i*th particle here is referred to as bird in the swarm and the *d*th dimension is the space.

X_{id} = This the position component of the *i*th particle in the *d*th dimension

c_1 = The individual learning such as the speed.

c_2 = The group learning such as movement and adjusting the speed and direction.

P_{id} = The social component of the P_{best} of the *i*th particle in the *d*th dimension. This is the individual particle best position in the *d*th dimension.

P_{gd} = The position component of the G_{best} in the *d*th dimension which is the position of the global particle best.

Rand () is the random fiction with numbers between [0, 1]

1. Convergence

$$V_{id}^{t+1} = K[V_{id}^{t} + c_1 \times Rand1 \times (P_{id} - X_{id}) + c_2 \times Rand2 \times (P_{gd} - X_{id})] \qquad (1)$$

$$K = \frac{2}{2-\varphi-\sqrt{\varphi^2-4k}}, \varphi = c_{1+}\, c_2, \varphi > 4$$

Equations 1 above explained the mathematical model behind convergence of birds. However, in order to understand not just the convergence but also the movement of these animals from place to place, orientation, repulsion and attraction were introduced. These will better enhance the understanding of these collective behaviours in these animals. Couzin et al (2002) explained these behaviours using a circle that shows the zor which stands for zone of repulsion, zoo which stands for zone of orientation and zoa which stands for zone of attraction as well as their field of view, which was shown as an angle $\alpha^{o} = 360^{o}$. Figure 2.3 below is the collective behaviour that was seen in swarm as adapted from Couzin et al (2002).

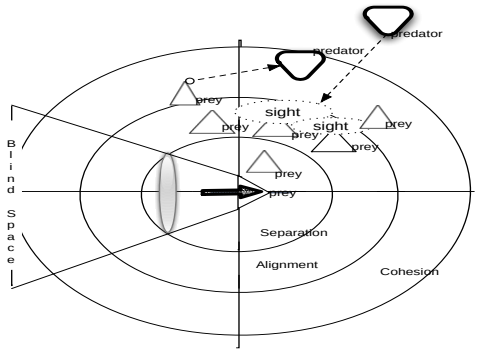


Figure 3-3 Preys and Predator Behaviours

Based on the diagram above in figure 2.3, the prey's interaction and communication are based on the closest six to seven neighbours, Ballerini et al (2008). However, their zone of separation is to maintain the distance between each other in order to avoid collision while moving in the same direction and attracted to each other. The diagram above in figure 3.3 shows that, the more the number of the observing preys, the more effective their detection capacity based on their position and field of view as shown also in figure 3.4. This shows that their eyes structure plays a very important role in detection of predators as well as their topology.

However, figure 3.4 below shows the six birds involved in the detection of predator and their field view as well as their networking and or communication between each other will trigger separation, alignment and cohesion. Most predators mainly have their eyes in front of their head while preys have their eyes on the side of their head. Thus, the eye structure gives the preys the peripheral view of the environment in detecting predators (Nguyen, Lee and Ngo, 1995). Each of these six birds in the group will only focus on the 60-degree angle of view instead of 360 degrees, which might cause distraction and strain. The 60 degrees will highly enhance their detection of predator.

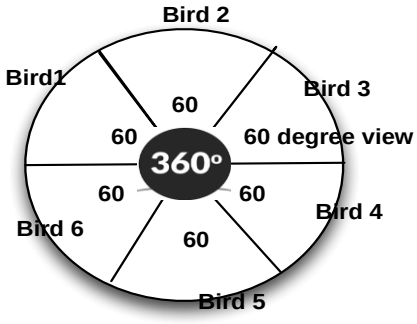


Figure 3-4 Birds Field View

3.6.1 Flocks Separation, Alignment and Cohesion

As stated by Couzin et al (2002) in a 3D space, birds in a flock tends to maintain their distance from each other in order to avoid collision or be exposed to predator which is represented in figure 3.3 and 3.4 above as separation or in other word repulsion. However, one of the reasons for their collective behaviour with regards to aligning themselves together is the attraction to each other or the risk of being isolated as stated by Partridge and Pitcher, (1982). This explains the cohesions as well as the alignment in these animals with regards to their collective behaviours. These three behaviours can be mathematically represented, but this project will only focus on separation as others are outside the scope of this study at this time, adapted from Couzin et al (2002).

1. Separation

$$d_s(t+\tau) = -\sum_{j\neq i}^{n_s} \frac{r_{ij}(t)}{|r_{ij}(t)|}, \qquad (2)$$

$$\begin{matrix} a11 & a12 & a13 \ldots\ldots a1n \\ a21 & a22 & a23 \ldots\ldots a2n \\ an1 & an2 & an3 \ldots\ldots ann \end{matrix}$$

Example 1

d_s = The domain of separation and changes in time t and spacing with regards to response latency.

Here j is not equal to i meaning that the elements of the diagonal are not to be calculated in the metrics. Where j=i are not to be included.

The elements of the diagonal are shaded in red in the example 1 above.

3.6.2 Scanning Frequency, Group Size and Topological Distance

There are some characteristics in birds that enhances their predator detection such as scanning frequency as stated by Dehn (1990); Pulliam (1973), group size as well as the topological

distance for their communication (Bialek et al 2012; Camperi et al 2012; Ballerini et al 2008). The scanning frequency in birds is dependent on the size of flock, Robert (1995) Lima & Drill (1990), Powell (1974). The increase in the size of the flocks will result in decrease in individual scanning and increase in security. However, Bellerini et al (2008), Demsar and Bajec (2014) demonstrated that it takes between six to seven birds in a group to detect any predator. Combining the group size and scanning frequency in order to answer the question of how detection of predator is done in birds by adapting Pulliam (1973) idea. Six to seven birds in a flock scan independently for predator in their environment. The probability of them escaping or fleeing from predator depends mostly on time. This is whether one of the six birds will detect the predator on time in order to allow them to escape or make alternative decision of fleeing the environment. The question now is, what is the probability of one detecting the predator. Mathematically, this can be illustrated with the following equation;

3. Detection

$$P_G = 1 - e^{-VTN} \quad (3)$$

P = probability

G = group

N = the size of the group (6-7)

T = time

V = scanning frequency

The probability of the group detection of predator is also dependent on the number of active scanning among the birds. The more the number, the less the angle of view and scanning, see figure 3.4.

$$V = \frac{-ln(1-P_G)}{T} \cdot \frac{1}{N} \quad (4)$$

$$V = -ln(1 - (1 - e^{-VTN}))$$

$$V = \frac{-ln(e^{-VTN})}{TN}$$

The equation (4) above shows the probability of 1 in a group detecting the predator based on the number of active scanning birds. The more the number of active scanning birds N, the more likely predators will be detected on time, the more protected they will be and less scanning duty for each bird. The more the number of birds N, the less their individual active scanning and angle of view as seen in figure 3.3 and 3.4 above.

3.6.3 Interaction and Communication Between Birds in the Flock

This study is particularly focused on European Starling (*Sturnus Vulgaris*), the way they interact in the flock and as well as their position. However, determining their structure or position during flight can prove difficult as the birds are constantly performing some aerodynamic movement and changing position in the sky (Pomeroy and Heppner, 1992). There are some factors that are important in determining the position of a bird during flight as well as their interaction. This will show how Starling in a flock interacts with each other. The following are the factors to be considered (a) topological as well metric distance, what is important? (b) Vision during flight (c) anisotropic and or isotropic factor.

3.6.3.1 Topological and Metric Rang:

Ballerini et al (2008) experimental research showed the important of using topological range instead of the popular metric range. Their experiment shows that metric distance does not guarantee robust convergence after predator attack. Examining the attack nature and the escaping strategies that made these birds to withstand any predator attack in such a large group. Although the two approaches, both the topological and metric distance guarantee cohesion. However, the question should be, under a strong predator attack like the one mentioned above, where three to five predators attacks the flock, how would you measure the resilient of the group cohesion? In order to answer this question, take a look again at Ballerini et al (2008) demonstration of system robustness using the mathematical approach bellow in 2D;

$$\vec{r}_{i(t+1)} = \vec{r}_{i(t)} + \vec{V}_{i(t+1)}$$

$$\Theta_{i(t+1)} = \frac{[\Theta_i(t) + \Sigma_j \Theta_{j(t)}]}{N_i + 1} \qquad (5)$$

$\vec{r_i}$ = The position of ith bird in the group

$\vec{V_i}$ = The velocity of the ith bird in the group

Θ_i = The bearing or the direction of the ith bird in the group

N_i = Number of neighbours interacting with neighbour i

In a topological range, a bird is only watchful of certain number of birds as they are moving which is $N_i = n_c$. The number of birds in topological range is certain while in metrics is based on the distance or metric range r_c. Analysis based on the simulation carried out by Ballerini et al, when a predator exert force or attacks the preys, the metrics range yielded a scattered bird of 24% while topological has only 0.7% of scattered birds. There is a tendency that the more scattered birds are in metrics range, they will lose touch with others in the group because they

are out of range, which is not the same with topological range because of its fixed nature. The result showed that there is a maximum probability of the birds in metric range breaking into 5 components after attack compared to 1 in topological range. Thus, in terms of resilience and stability of a system, topological range is preferred against metrics. As birds' scatters, they expose themselves to predator, which made it easy for a predator to target and catch a single bird than a group of birds. The predator approach always changes the direction of the movement which goes away as the predator distance goes farther away from the preys. This shows that the heading or bearing Θ_i of the prey's changes with every introduction or appearance of predator and stable as the predator distance decays both in metrics $(1/r)$ or topological range $(1/n)$. The equation below is as a result of force introduced by the predator presence to the preys that brought about change of bearing and heading in a 2D form.

$$F_0 \frac{[y_i \cos(\theta_i) - x_i \sin(\theta_i)]}{r_i^2} \qquad (6)$$

3.6.3.2 Starling's (Sturnus vulgaris) Vision During Flight

Martin (1986) explanation of bird's vision based on their experiment of the eye structure of the starling birds *(Sturnus vulgaris).* The experiments showed that starlings have a blind rear axis as well as lateral visual axis. This explains why birds can only keep track of certain number of birds on both sides during foraging or flight. They can move their eyes together as well as independently without turning their heads (Lawler, 1993 PP. 61). Their vision is one of their survival techniques as well as interaction when something is detected. They watch the movement of others within their vision field with both left and right eyes to see the direction of their movement and what they are doing (Ballerini et al, 2008). However, their detection and interaction are not mainly vision focused, rather it contributes to their survival. Focusing only on vision might be detrimental to the species since foraging and sleeping limits vision (Devereux et al, 2005).

3.6.3.3 Anisotropic and Isotropic Factor:

Ballerini et al, (2008) hypothesised that the structure of each bird or the flock might be a strong determinant of its anisotropic nature during flight. Starlings have a lateral visual axis as well as binocular visual capability and they can only interact with a certain number of individuals at a time. Their eyes structure allows them to interact with the birds by their sides or within their filed view. There are some restrictions in their view with regards to their eyes structure

such as not able to see the birds at some point at the back as well as front. This showed that they could only interact with birds at some positions by their sides. The empirical studies carried out by Cavagna et al (2010) showed that anisotropic evidences in the angler distribution is as a result of interaction between birds in the flock. However, they pointed out that if the flocking conditions were non-interacting, the distribution would be isotropic. The idea was supported by (Ballerini et al, 2008). This explained that the more birds are interacting together, they flock together and move together or their direction of turning as well as movement will be determined based on their interacting companions. In a situation where they are not interacting, movement can be uniformly but in all direction. Hence, Cavagna, et al has developed some mathematical models that determine the anisotropy of birds in a larger group titled "Anisotropy Matrix" as seen below.

$$M_{\alpha,\beta}^{(n)} = \frac{1}{N} \sum_i^N \upsilon_i^\alpha \upsilon_i^\beta \qquad (7)$$

M = Represents the sum of many projector such as direction of the nearest *n*th neighbour i

υ_i = The normalised distance vector of the *n*th nearest neighbour of bird *i*

In order to quantify the anisotropy in neighbours' distribution and see how it decays with the increase in density, some properties will be considered such as eigenvector as well as eigenvalue. The eigenvalues of M can be represented as $\lambda 1 < \lambda 2 < \lambda 3$ while the eigenvectors could be represented as W_1, W_2, W_3. In this instance, the eigenvector W_1 represents where the nearest neighbour is not likely to be found while W_3 represents where the nearest neighbour is likely to be found in the flock. Based on further analysis from Cavagna et al (2010) on their projections to determine the degree of anisotropy, which shows that the nearest neighbour was not found in the direction of motion V that made the value on the P $(\cos(\theta))$ (X axis of the graph which begins from -1 to +1) approximately1. However, the scalar value is less as well as anisotropy when the number of neighbors increases. Hence, the degree of anisotropy can be found based on this experiment by using the square scalar value such as $\gamma_1(n) = (W_1.V)^2$. This can only be calculated when the individuals are interacting in the flocks which is anisotropy but in the case of isotropy, the eigenvectors are statistically uncorrelated with the direction of motion V, since there is no previous known direction. Based on the fact that this study is considering only integrating the degree of anisotropy in determining the interaction between birds in groups n_c or their interaction range, isotropy will not be further evaluated at this time because it deals with uniformity in all direction (three dimensions $\gamma =$

3 α), but this study is dealing with flocking behavior. Moreover, this study recognizes that in groups or flocks, there are birds at the border, which only have neighbors on one side, as well as birds in the front or back of the flock, which has neighbors only at the back, and in the front. The interaction can only be effective and complete when the neighbors are complete, in this case n_c = 6.

3.7 Reason for Birds Approach in Intrusion Detection System

Cavagna et al. (2012) in their statistical mechanics modelled the collective behaviours seen in the flocks of bird using maximum entropy model. This approach and model in order to describe how to capture the movements of birds in a program. The behaviour of these birds was modelled, such as movements in the flock, position and direction of movement of a single bird in the flock as well as the whole flock. Through the observation and some experiments conducted about the flocks of birds by Ballerini et al (2008), suggested that interaction is based on topological view rather than popular metric view. The study was titled "Interaction ruling animal collective behaviour depends on topological rather than metric distance". However, Cavagna et al (2012) Maximum entropy was used to represent the structure mathematically for a better representation. The representation encompasses both individual bird movement as well as groups from one position to the other. However, it clearly shows how information travel from one bird to the entire flocks, named in their model "free correlation". Birds in the flock monitors the movements of their neighbours, they can only monitor around six to seven birds at a time as stated in their experiment. This explains also the Chorus-line hypothesis and the spontaneous movements of the flock. The movement and monitoring of one another are the information that will be useful in modelling the approach in this research as seen in previous sections as well as this section.

The chorus-line hypothesis was as a result of monitoring the movement of avian flock by Potts (1984). This Transends the way the birds move in a coordinated manner at one time during flight. The observation of these birds was named here by Potts as chorus-line and at the same time, this is how information flows from one bird to the entire flock. How does bird at the boarder or in the middle know that the movement will be left or right, up or down? This is what Cavagna et al called free coloration of information. This is one of the reasons for exploring the use of flocks of bird's approach in security. These animals are not necessary very intelligent, but they can be in thousand and even million in the flock, but it is harder to catch one in such large group. Each of them is capable of detecting a predator because they do not necessarily

have a leader. They coordinate and corroborate with one another in such a uniformed way. This kind of distributed detection approach worth emulating for securing ICS.

3.7.1 Prey Predator Approach in Birds

From the predator prey perspective, predators are bigger than preys as stated by Rosen and Hedenstrom (2000). Their experiments with Falcon and seven different preys showed that preys in their characteristics always develop escaping strategy. Since predators are bigger and stronger, their wings and strength are stronger. This shows that they can fly faster than the preys. This is one of the reasons a Falcon can target a lone European starling and catch it as they fly on a straight line. However, the escaping strategy they develop is mostly based on the previous experience. Preys are smaller, and their turning gambit is smaller while that of the bigger birds are more. This is an escaping strategy most preys employ for keeping away predators or escaping from predators.

There is a question that arose in this study which prompted in investigating this approach seen in the flock of birds. The question is as follow; what if the flock is faced with multiple predators such as 5 and more predators? This question occupied this part of research. In order to answer this question, an experiment was carried out in a village in Germany in August 2016. The experiments were with some flock of European starlings as they are the particular birds available there. Three small round balls were used for this purpose. The balls were thrown in the mist of these birds over several times and the same results were received. The throwing was to emulate the attack from the predators to the preys and to see the reaction of the preys when attacked by these predators. However, the result was as seen in figure 3.5 and 3.6. This was a demonstration of multiple predators' attack of the flocks of bird. These birds in the flock react the same way they will react to one predator attack.

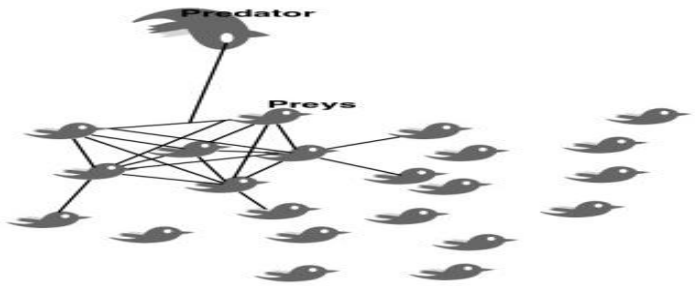


Figure 3-5 Prey Predator Detection and Prey Communication

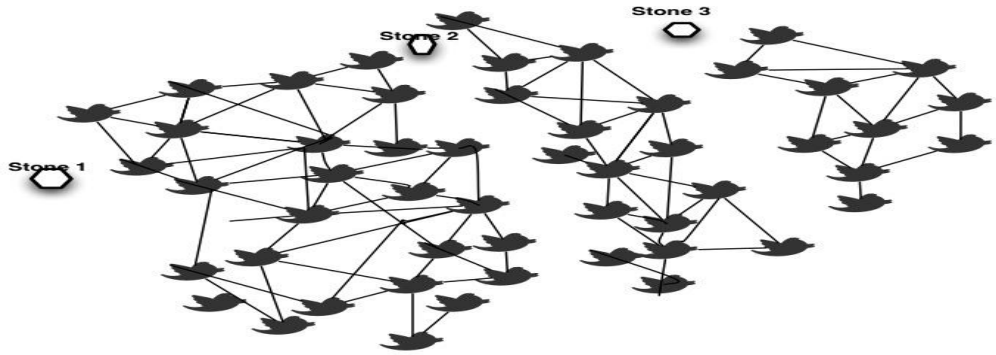


Figure 3-6 Ball Stone to the flock and the results

This behaviour seen in the flock of birds as demonstrated in figure 3.5 and 3.6 might be explained from the perspective of many eye hypotheses from Rosen and Hedenstrom (2000). Because each bird has two eyes and many of them have many eyes. When many eyes are watching, it is easier to detect any incoming predator. The way birds detect predator is based on their scanning frequency as stated above by Glueck, (1987). Vigilance increases as a result of previous experience. The increase in scanning frequency will determine whether the bird will detect the predator. As a group, the more the number, the less the individual scanning frequency. In terms of starling, they have eyes for detecting predator, the eyes for their type of food as stated by Martin, (1986). This extraordinary detection approach in the flocks of bird need to be emulated, as each bird is capable of detecting a predator. One of the pressing issues nowadays is designing and building a detection engine that can handle big data.

Eslamnezhad and Varjani (2014) solved the issue of big data processing using MapReduce and Hadoop framework. The integration of their algorithm in the cloud resolved the issue of data processing and clustering. Suthaharan (2013) also proposed the use of Hadoop framework in IDS for reducing big data challenges. Tsang and Kwong (2005) used different agent at different levels in the control station. Their approach was distributed in solving big data challenges. Based on this information, Hadoop framework is a better option for resolving big data challenges with regards to data processing and storage. This means that data will be processed faster without thinking about the memory or computer processing power.

3.7.2 Hadoop Framework

Hadoop framework is a framework that has attracted wider research and development in resent time. Hadoop is a framework from apache that was designed by Doug Cutting and Mike Cafarella. It is designed with a distributed file system for running, storing and processing big data, Lam (2011, P. 4). The increase in the amount of data generated daily from all sources of

life inspired the idea. The initial idea was first conceived and inspired by a white paper published by Google on MapReduce as stated by Lam (2011).

Hadoop framework is made up of MapReduce and the Hadoop Distributed File System (HDFS). The MapReduce is for the processing of data while HDFS is for data storage. There are distinctive characteristics that distinguished Hadoop from a database and other frameworks that uses distributed systems such SETI@home and these distinguishing factors are; Robustness, Scalability, Simple and Accessibility. The robustness of Hadoop can be seen in the way failures are been handled. Due to the fact that it runs on commodity hardware, failures are being anticipated and managed. Hadoop is scalable based on its architecture. It scales linearly and can handle bigger amount of data faster by simply adding more nodes. The simplicity of Hadoop made it easy to write your own code and integrate it to the framework. Many applications nowadays are running on Hadoop due to its simplicity. it can be accessible everywhere if it's integrated to the cloud environment (Lam, 2011) as seen in the architecture in figure 3.7.

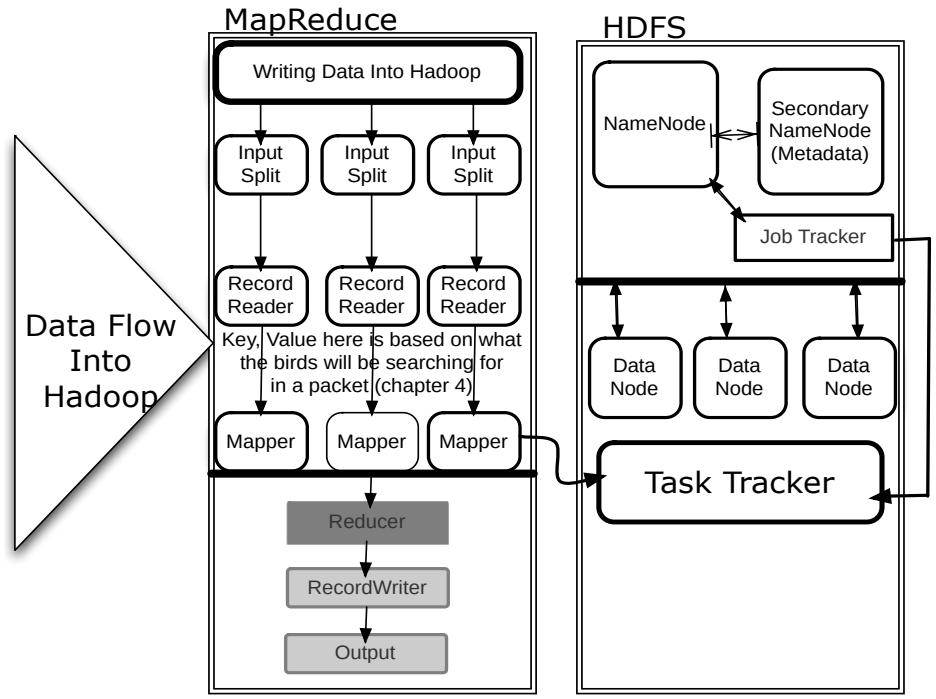


Figure 3-7 Hadoop Architecture

The architecture above in figure 3.7 is the Hadoop framework. The data that are coming in or going out of Hadoop is processed by MapReduce and stored in the HDFS. This goes as follows; client will contact the NameNode with the amount of data it would like to store in Hadoop. The NameNode will now talk to the DataNode based on the metadata that was available, the

NameNode knows the DataNode that has free spaces. The NameNode will now assign nodes where the data will be stored and inform the clients. Based on the Hadoop architecture, data are stored in chunks of 64mb and up to 128mb. Hadoop is good for big data storage and processing because of the 64mb chunks or 128mb depending on the configuration (Apache Hadoop, 2014).

When data are stored in HDFS, it is processed using MapReduce. These are unstructured data at this time. If the client contacts the NameNode for data, the NameNode will contact the JobTracker. The JobTracker will get all the information about the data from the metadata such as; location of the data. The processing of the Job will start by splitting the data as stated in figure 3.7 "inputSplit". The data will be split in 64mb chunk, depending on the type of data type. The number of splits will determine the number of mapper for the data. Before passing the data to the mapper, it will be converted into key value pair. In order to convert it, the recordReader will be needed for every InputSplit and Mapper. The conversion of these data depends on the format of the data. The RecordReader will read one line of the data at a time and convert it into key value pair. There are three main file formats in Hadoop such as; TextInputFormat, KeyValueTextInputFormat, SequenceFileInputFileFormat. However, by default, files are converted to TextInputFileFormat when there is no file format specified or available. When the RecordReader reads a line, it covert it to Key Value pair and push it to a mapper and go back and read the next line. The mapper at this time will store the key value pairs and the words or alphabets based on the InputSplit. The input and output type can be any of the java-Hadoop primitive type such as IntWritable, Text, LongWritable and among others. The mapper now processes these inputs and output values and forwards it to the Reducer. The Reducer will now combine all the available key value pair's data. This data is the combination of the processed job from Mapper and Reducer, which is called intermediate data. The Reducer will now carry out the work of sorting and shuffling. The shuffling is where all the duplicate keys will be combined and sorted as one value. The value now will be pushed to RecordWriter, which will now push that as output directory (Durgasoft, 2014) for storage and other things.

Hadoop is a framework that has other software application that are running on it such as HBase as a database for data storage, Hive for analysis of the dataset that are stored in HDFS, Pig which was initiated by yahoo as a programing language, Zookeeper which is for data coordinating in distributed file system (Apache Hadoop, 2014).

Summary

Hortonworks has done quite a lot of work in the area of data streaming with some of the apache software. There are number of tutorials and projects from them online that one can easily follow. The similarity between some of their work and the work in this project is the real time streaming and the application of Hadoop framework. These were evidence in the case of truck tracking in the real time (Hortonworks, 2017). Their work reports the speed of the vehicle as well as the location and activities in real time. Even the analysis of the real time toll and traffic that was conducted by Hagan (2017) using Hortonworks. The algorithm that were used for these analyses was placed in the cloud and use for the real time detection of anomalies.

This section of the project reviewed both the current and past literatures on the intrusion detection systems in order to find similarities to the proposed work. The IDS was reviewed both the state of the art and the future IDS which shows the trend and the paradigm shift. The trend showed that the future of the IDS has shifted from signature based to anomaly detection as seen in the case of Tsang and Kwong (2005) as well as others. This is a paradigm shift even for the ICS as the number of attacks on these critical infrastructures are on the increase based on the survey from Dell and NIST. The anomalies detection approaches are also shifting towards emulating biological means such as immune system, operation of Ants, Bees and Birds. They can detect both known and unknown anomalies depending on the strength of the algorithm that control the detection engine.

The exploratory studies carried out on the flocks of birds and their characteristic behaviours was very interesting. One of the findings was the flocks of birds' collective behaviour in detecting predator and their aerodynamic movement in the air. Birds in the flock are hard to catch compared to a lone bird as hypothesise by Martin, (1986). Bird such as European starling detects predator based on topological view as stated above in the review. This shows that the detection and movement is based on the number of birds in their view. Birds move based on their neighbours' movement and the more they are in the flock, the less the level of vigilance for individual bird and increase in detection. They do not have leaders but each of them can detect a predator which is distributed in nature. Hence collective behaviour can be seen in the way they respond to attack as a flock. This was tested by using some small balls and the analysis proved that the same way they respond to a single attack, they respond to multiple attacks. This was one of the reasons for chosen their approach for the detection of anomalies on ICS.

As many of the biological entities are now using IDS, such as Bees called artificial bee colony, artificial immune for IDS in securing ICS as described above. Bird approach has been proposed by Okeke and Blyth (2016), Okeke and Blyth (2017) for securing ICS. Each of these animals has the capacity of detecting predator and move at the same speed as others. Their detection in the flock is based on their neighbour's movement and this formed their information transfer. As explained by Cavagna et al (2012), the information transfer is named "free coloration" of information, which allows the birds to move the way they move in the group. Their aerodynamic movement is based on the movement of their neighbours. Some of these birds might not see the predator, but when the neighbours are moving, whether sideways or forward, they will follow the suit. They are maintaining the flocking behaviours of separation, cohesion and attraction. These three characteristics made them flock together.

In summary of all, three steps have been indicated for use in this project. The first is the modelling of the flocks of birds based on the characteristics that were identified. The second is the solution for the big data challenges using cloud and Hadoop framework. The third is the SCADA system that will be used for data collection and testing. The function of Hadoop has been studied and its viability for this work. Based on the available information, Hadoop will be used for solving big data issue as well as HBase for the database.

CHAPTER IV

4 SYSTEM DESIGN AND MODEL

Designing the system is a process that will show how the system will be put together. This will be handy in virtualising various components of the system before it is finally created or modelled. Designing a system is like a map that will guide someone through in arriving at research hypothesis or answering research questions. It provides a clear path to follow when building or modelling the system (Maxwell, 2000). However, there are laid down principles in research as stated by Seale, (2006 p.130). These principles are to make designing and answering research questions easy and to recognise any issue that might occur in the process. Below are some of the principles that might be needed for the purpose of this research as seen in research methodology in the appendix;

1. You will need clear research question/s and or a generated hypothesis
2. You should adopt a method and or tools that will yield robust data analysis in order to meet the research aims and objectives.
3. You should consider ethical issues when selecting and using research approaches.

All of the above information is very vital in research but not all will be followed in designing this system. The research hypothesis was made clear in the beginning and simplified as well as the research aim and objectives. The research methodology was chosen after a careful research and analysis, please see appendix. The approach will definitely yield a robust data analysis that will aid in achieving the objectives of the research. Clearing the ground for the design of the system involves gathering the necessary requirements. These requirements will show the understanding of the systems that is about to be designed. After the requirement gathering for the system, the architectural design of the system will follow. The architecture will be based on the requirements that was gathered as well as aligning the project to the FINER criteria of Design Science (DS) research methodology, please see appendix. Designing the system will also later involve the introduction of the case study and all the components of it. The case study will involve the platform where the model will be executed. The platform is the locomotive test rig equipped with some vital components of SCADA system. Designing and experimenting with the system will help in locating all the vital information that is needed to model the system as well as the possibility of integrating the birds of prey model into the system. This will show the data collection points as well as the integration point for the model.

Data collection is vital because it will determine the type of data the model will be looking for in the environment. The environment or the SCADA system will be built with two type of Programable Logic Control (PLC), the Siemens and Schneider Electric. These two companies operate two distinctive types of communication protocol. Hence these protocols need to be identified for the model as well as other vital information such as data flow frequency and among others. However, other protocols and their structures will be explained and explored. The problems the proposed project will solve will be listed as well as the benefits.

4.1 The Proposed Approach and Big Data

The proposed approach is the use of flocks of bird approach to predator for the detection of anomalies on ICS. Okeke and Blyth (2017) proposed the emulation of the distributed detection seen in flocks of bird for securing ICS. They were of the opinion that the detection approach by the flock of birds is distributed in nature. Any bird in the flock can detect a predator and in the case of multiple predators, there will be multiple parallel detections as seen in figure 3.6. Flock of bird approach in detecting predator is distributed in nature which is suitable for detecting predator in big data environment. However, the distributed nature of the bird's prey in detecting predator might be enough in handling big data issues such as storage and data processing. There are other issues to consider such as the computer power for processing the job, SCADA protocols from different vendors. These will be considered during the design of the system.

4.2 SCADA Protocols

Protocol is a communication language that allows devices and equipment's to communicate with each other in an environment, Kalapatapu, (2004). In an industrial system such as SCADA, the RTU communicates with the central or master station through protocol. The RTU and PLC are built with CPU and memory embedded to it, which allows them to process, and retains data for some time. However, SCADA communication between software and hardware's has been an issue in the past. Due to the fact that vendors produce their own hardware and software, any other software or hardware that is not from that particular company might not be compatible. International Standard Organisation (ISO) resolved this issue by the introduction of Open System Interconnection (OSI).

Clarke and Reynders (2004 p. 5) in their book explained that ISO faced with the challenges of getting the systems to work together, which brought about the introduction of ISO reference

7498 as a framework. The ISO7498 was introduces in the late seventies as the OSI for easy communication between vendors with regards to their products. This resolved the issue of compatibilities between vendors product, thus by conforming to the OSI. The framework defines the operation and function of each of the seven layers of the OSI for communication flow and compatibility. These seven layers are organised and structured in a way that they communicate with each other, which is now widely adopted by most vendors. Please see below in figure 4.1 the diagrammatically representation of the seven OSI layers.

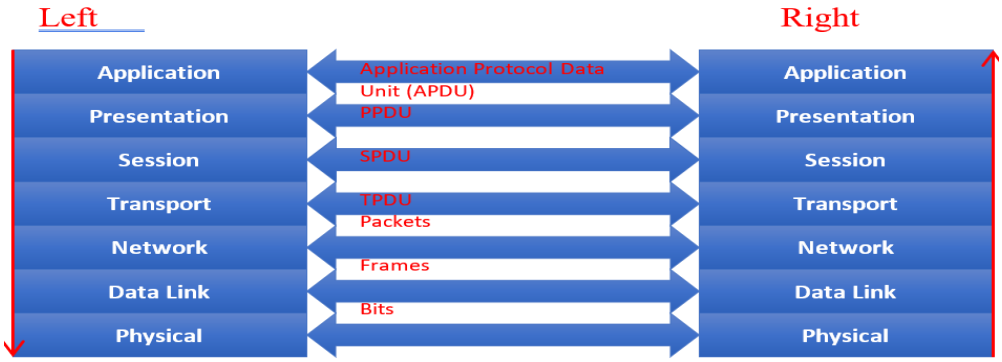


Figure 4-1 Communication Architecture of the OSI Model

Figure 4.1 above showed the 7 layers of the OSI model of communication (Mehta, and Sahai, 2017). The left side on the diagram is the transmitting side while the right side is the receiving side. These 7 layers are interconnected to each other in a system. Information is passed on to another layer through Service Access Point (SAP). This is the point where one layer passed information to another layer in the OSI model. When transmitting information, the layer that is transmitting will add extra information such as header and or protocol control information before transmitting to the next layer. This depends on the position on the OSI model, because the further down the transmission goes, the more header is attached to the data. Thus, this is called Protocol Data Unit (PDU) as seen in figure 4.1 above. The header contains information that is needed at the receiving end with regards to destination and function. The physical layer here will receive the packet and convert them into electric signal of 0s and 1s and convert it back to packet before transmitting it to Data Link. On the transmitting side, the information is received from Data Link by physical layer as packets with header, service and function codes. This packet will be converted into electrical signals before transmitting it to the receiving physical layer on the other device. When the packet or the bits as seen in figure 4.1 is received, the information will be retrieved and converted back to packet. On the receiving side (the right side), the packets are stripped off the header information until it gets to the destination. The transmitting end adds header information while the receiving ends remove the header until it's only the data or message remains (Clarke and Reynders, 2004 p. 58).

4.2.1 Modbus Protocol

Modbus protocol belongs to the group of protocols that were created with the OSI conformity for easy interoperability among vendors (Deng, Peng, and Liu, 2017). This industrial protocol has been in use for many years. The protocol has ASCII, TCP and RTU types, which makes use of RS232 or RS485 for their physical connectivity. These are the serial communication connection port that allows flow of information. Modbus protocol is made up of function codes, which defines the request and the encoding scheme for data transfer. The encoding scheme can either be a 1-bit coil or a 16-bit register data. The protocol was originally created by Modicon in 1979 and later took over by Schneider Electric. Modbus operates master slave with one master and many slaves depending on the configuration. (Zhi-Qiang and Yu-lin, 2009).

Modbus as an application layer protocol designed as a master slave communication protocol (Modbus, 2006). Its services are based on function codes such as read or write from register or coil. When a function code is sent out from a client to the receiver, it tells the receiver the action that client would like to initiate if the function code is valid. The valid function codes are from 1-255 while some of these codes (128-255) are reserved for the purpose of exception. The request from a client contains not only the function code; there are other important information that will allow the receiver to locate the right information. The information contained might include the following but not limited to this only; the register address, the amount of thing to do, the count of the available data bytes in that field. If the information provided by the client are complete, the receiver will echo back the request based on client request. However, if the information provided such as function code is not correct, the receiver will respond with exception code. The communication between the sender and the receiver are based on the normal handshake; the sender sends a request to the receiver and the receiver will check the code as well as the data and if it meets the request, it will carry out the request accordingly. If the codes are wrong, exception will be sent with the code also. The Protocol Data Unit (PDU) has a data limit of 256 bytes for Modbus based serial communication. The 256 bytes is further reduced by removing the receiver address which is 1 byte and the Cyclic Redundancy Check (CRC) which is 2 bytes. Removing these will result in 253 bytes remaining, which is the real size of the PDU serial communication. TCP is a bit different; the Application Data Unit (ADU) is 260 bytes minus the 7 bytes of the Modbus Application Protocol (MDAP). However, both the serial and the TCP communication lines amounts to 253 bytes after the deductions (Mohagheghi, Stoupis and Wang, 2009; Zurawski, 2015 Chapter 10).

The data types that are supported by Modbus are; Discrete Inputs, Coils or Discrete outputs, Input Register, and Holding Register, (Zurawski, 2015). Table 4.1 is the Modbus data types

Data Types	Object Type	Type	Information
Discrete Input	1-bit	Read-only	I/O system data as an example
Discrete Output or Coils	1-bit	Read-Write	It is a read write data, therefore an application program can read from or write to it.
Input Registers	16-bit word	Read-Only	I/O system data as an example
Holding Registers	16-bit word	Read-Write	It is a read write data, therefore an application program can read from or write to it.

Table 4-1 Modbus Protocol Data Types

4.2.1.1 Protocol Description:

The Modbus Protocol consists of PDU, which is made up of function code and the data as explained in Modbus (2006). The function code is made up of 1 byte of data while the data field being sent by the client consists of 1-byte address field, which is at the beginning of the Application Data Unit (ADU) and the 2 bytes of error checking field. Figure 4.2 is a typical Modbus protocol ADU and PDU.

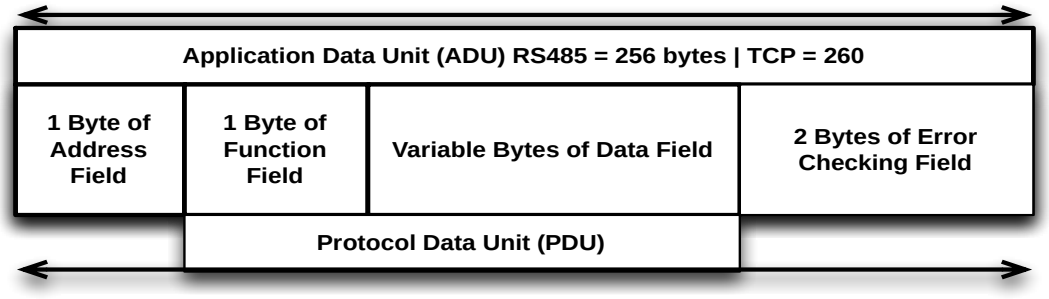


Figure 4-2 Modbus Protocol Structure

The address field contains 1 byte of the slave address. When a slave responds to the master, it adds its address for the master to recognise the origin of the response. When the master send

message to the slave, it puts the slave address for the destination purposes. This address is in the range of 0-247 decimal where 0 represents broadcast address. However, function codes are very crucial for the message transmission. On like the address field of the frame, the function code is made up of 1 byte, which is either two ASCII or 8-bit characters. These codes are in the range of 1-255 decimal, which some of them are reserved. Some companies use these reserved codes for specific purposes, which are user defined. However, function code 43 is used for a specific purpose, which is for device identification. This code has its Modbus Encapsulation Interface Type, which can either be 13 or 14. These two codes belonged to the reserved function codes. The function code will tell the slave the action to be taken. The actions can either be to read, write or to perform both. The data field on the other hand contains two hexadecimal digits. The primary purpose of it is to provide additional information required by the slave in order to complete the action in the function code. For example, if the function code was to read, the data field will contain information on the register to read from as well as the beginning and the end of the register address. The data field can also contain zero information in the case where only function code is sufficient in fulfilling the task. The error-checking field contains two bytes of information depending on the type or error checking that is applicable. There are two types of error checking that is applicable on Modbus protocol namely; Longitudinal Redundancy Check (LRC) and Cyclic Redundancy Check (CRC). The LRC mostly performed when ASCII mode is in use with regards to character framing while CRC is widely used in RTU mode. This as seen in figure 4.1 contains 2bytes of data, which is appended at the end of the message (Wright et al, 2004 P.98-115).

4.2.1.2 Modbus Function codes:

Control Codes	Function Code	Hexadecimal
Read Coil	01	01
Read Contact/Discrete input	02	02
Read holding register	03	03
Read input register	04	04
Force Coil/Write single coil	05	05
Load Register/Write single register	06	06
Read Exception status	07	07
Diagnostic (this has sub codes)	08	08
Get Com event counter	11	OB

Get Com Event Log	12	0C
Force multiple coils/Write multiple coils	15	0F
Load multiple registers/write multiple registers	16	10
Report Server ID	17	11
Read general references/Read File record	20	14
Write general references/Write File record	21	15
Read Device Identification (MEI 14)	43	2B
Encapsulated Interface Transport (13, 14)	43	2B

Table 4-2 Modbus Function codes and their Hexadecimal values

Table 4.2 above is the Modbus function code available and their hexadecimal values (Modbus, 2006). The function code contains the information about the action to take, whether to read, write or hold information. The first function code on the table 4.2 was 01, which is to read coil. There are quite a number of coils in a remote device such as RTU and this information will carry along the data field, some required information. Thus, it will specify the starting coil and the end coil to be read. The first coil in the PDU is 0 instead of 1. This means that the numbering starts from 0 instead of 1. Therefore, when the information is to be read or write to coils 1-10, it will write to coil 0 to 9. Below is the example of request to read coil with function code 01 and its response (Wright et al, 2004).

Name of the Field	**Hexadecimal Value**
Address	06
Function code	01
Starting address or Initial coil offset Hi	00
Starting address or Initial coil offset Lo	C1
Number of points Hi	00
Number of points Lo	07
Error Check (CRC/LRC)	F9

Table 4-3 Request to read coil (Function code 01)

Table 4.3 is the request to read coil frame that will be sent out (Gavazzi, 2010). This includes the function code to read coil, which is 01 and the starting address or the offset into the coil.

The number of points or the data length depending on how it is viewed represents the number of coils that would be read from.

Name of the Field	Hexadecimal Value
Address	06
Function code	01
Byte Count	01
Data (Read coils 201…. 194)	AC = 10101100
Error Check (CRC/LRC)	F9

Table 4-4 Response to read coil (function code 01)

Table 4.4 is a response from the request in table 4.3 to read coils 201 to 194 which has a hexadecimal value of AC. Translating the hex value of AC to binary, it shows binary of 10101100 from the Most Significant Bit (MSB) to Less Significant Bit (LSB). The coil 201 is the MSB while 194 is the LSB starting from left to right as shown in table 4.3. Thus, the binary representation of 10101100 of the bits is ON OFF ON OFF ON ON OFF OFF. However, if the binary representation or the bits starts with 0 from the MSB, the count will start from the position of the first 1 and the initial 0s will not be counted. To make up for the remaining byte that were either removed, it will be padded at the back. For example, the binary value of hex 26 is 00011010. This will be represented as ON ON OFF ON OFF. This might be from coils 89…71. Thus, the MSB 89 is in the fourth bit position while the 71 input is the LSB (Clark and Reynders, 2004 P. 45-55). This can also be represented in a flowchart as seen in figure 4.3.

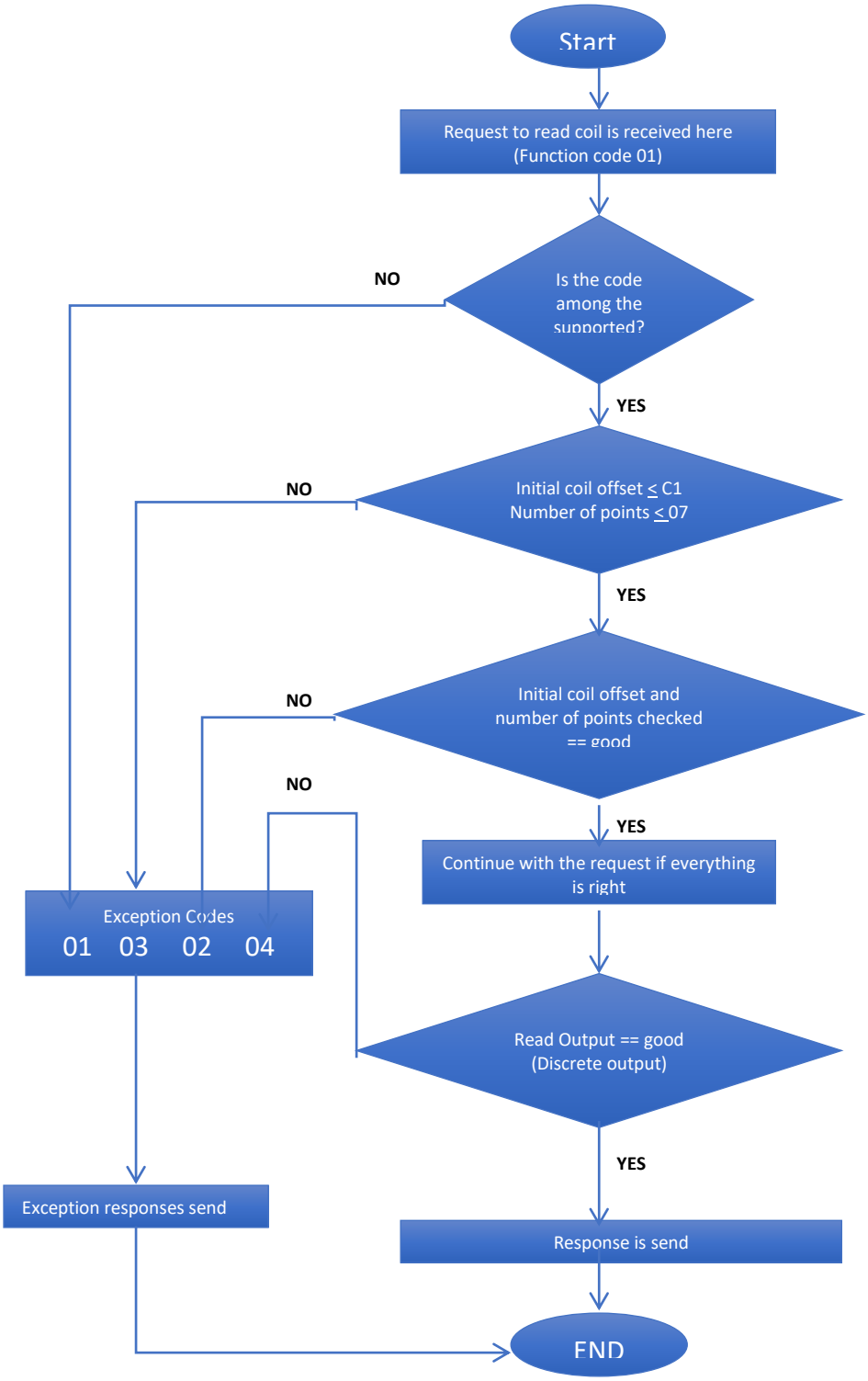


Figure 4-3 Request and Response for Function Code 01

In a general point of view of the Modbus protocol function code processing. When a request is made, it's only two things that can happen; it is either the process runs through or it returns an exception. If the request is right and all the parameters were met, the request will be processed, and positive response will be received, if it was meant to send one. However, if the expectations

were not met, it will return exception information to the client and the reason for the error (Modbus, 2006).

Modbus as a communication protocol is an application layer protocol, whether it is Modbus RTU or TCP. Modbus RTU is a master/slave protocol for Electronic Industries Association/ Telecommunication Industries Association (EIA/TIA) 232 and EIA/TIA-485, while Modbus TCP is an Ethernet protocol. When a packet is sent, the receiver does not really know the content of the packet. The reason for protocol is to help the receiver process the packets. The Function code will tell the receiver what the sender or the client would like to do. The Modbus TCP uses the IP address in locating the sender or the receiver while Modbus RTU uses device identification (ID) for device identification. This information is very important in identifying the communication protocol on a device on industrial control system (Cena, Bertolotti, Hu, & Valenzano, 2014).

However, this research will go further in introducing and explaining other popular available open protocols. The introduction will focus mainly on their data representation. The data representation of each protocol will show the data structure and the available representation. The next available popular protocol is the Distributed Network Protocol version 3 (DNP3) and its data representation will be introduced.

4.2.2 DNP3 Protocol

DNP3 (Distributed Network Protocol version 3) protocol was originally developed by Westronic and later by Harris Controls in 1990 as documented by East et al (2009 p.67-81). The protocol was basically developed for electrical industry, which is more widely used in North America and Asia. The protocol adopted the open standard format for interoperability between vendors. However, the frame format was from one of the International Electrotechnical Commission (IEC) 870, which are Frame format (FT)1.1, 1.2, 2, 3 and others. This is a communication protocol that was designed for communication between master and slave or master and the RTU. The communication can either be one master with multiple slaves or two masters with one slave or multiple slaves depending on the system design. Only the master station can send for a request although slaves are allowed to initiate communication but not request. The protocol uses a periodic poll for failure detection in communication in the system. Slaves normally send responses to the master station whenever there is a change to the system, which the master uses to know their state. Its uses Enhanced Performance Architecture

(EPA) mode for its communication which was introduced by the International Electrotechnical Commission (IEC). This was only developed in order to simplify the OSI framework communication for better interoperability. These are three layers model instead of 7 that is in the OSI. It comprises of two hardware layers, which are physical and Data link layers and one software, which is Application layer.

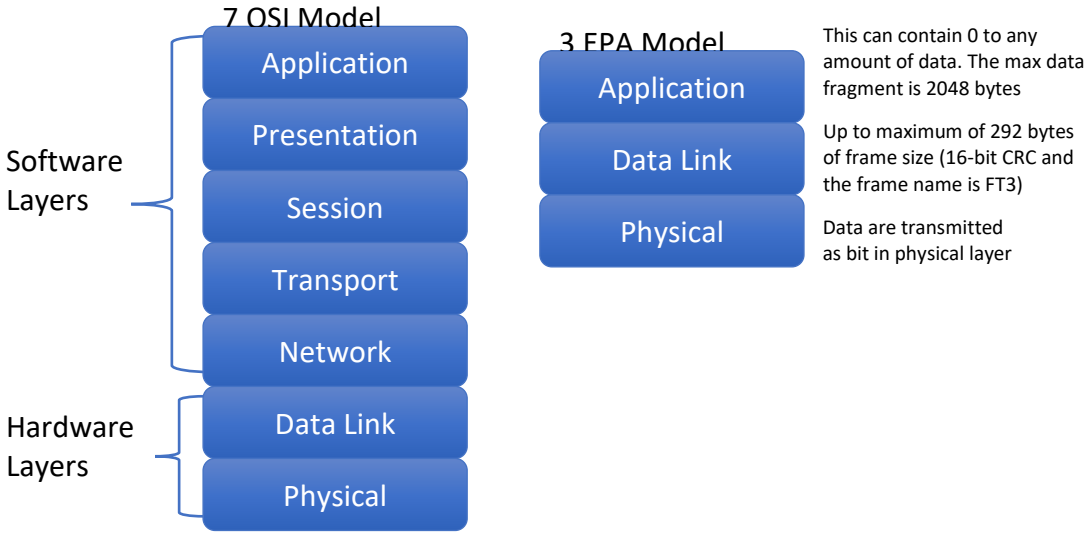


Figure 4-4 OSI and EPA model for DNP3 Communication

The information in figure 4.4 was curled from Clarke and Reynders (2004 p. 73-81). The introduction of the three layers EAP has made the communication easier as the communication would not have to go through all the seven layers. The application layer in DNP3 can be of any size, which is made up of Application Service Data Unit (ASDU). The ASDU will help in managing the data in blocks initially and then adds header into it in order to form the APDUs depending on the size. This header will carry the control information and it is referred to as Application Protocol Control Information (APCI). The size of the ASDU will determine the number of the APDU that will be created for the next process. The maximum size of this APDU or normally termed fragments in case of more than one is 2048 bytes. However, the request and response data are not the same with regards to the frame. The request frame header is made up of application control and function code while the response header adds extra internal indication bit to the data. This data will then be handed over to the transport layer which will then brake the data down into smaller sizes. The data is broken down to a maximum of 249 bytes with 1-byte header making it 250 bytes. The transport layer receives the data and prepare it for the next level by adding additional 10 bytes of header to it. This header is made up of 16-bit error checking code and the maximum size of the frame is 292 bytes. The transmitted frame

will have the finish (FIN) when the transmission is finish or first (FIR) when it is the first to be transmitted and a 6-bit sequence header added to the frame. The physical layer will then now convert the data into bits for further transportation. As stated above, DNP3 adopted the FT3 frame and below is the frame format (Clarke and Reynders, 2004).

Table 4.5 is a representation of FT3 frame used by DNP3 for its request and response messages. The frame begins with 2 bytes of start or sync data that identifies the source of the information, followed by 2 bytes of the length. The length specifies the length of the message, which will show the number of octets contained excluding the CRC. The control or the link control, which provides data flow controls (sending and receiving data activities) through the link layer. This will identify whether it is receiving or sending frame. Depending on the type of message, whether secondary or primary or a message from primary to secondary or vice versa. There is additional information that is contained in the message such as the direction (DIR), Primary, Frame count bit, Frame count bit valid, Reserved, Data flow control bit, and function code. The source and destination addresses are for the source and destination addresses which starts with the Less Significant Bit (LSB) to the Most Significant Bit (MSB) respectively. The user data will contain 16 bytes of user data while the CRC contains 2 bytes of error checking code for any malfunction or misrepresentation of the message. Whether the message is from the secondary or primary source, the message has a format it should follow. The function code will describe the message action such as reset link (for primary P0 and secondary S0), which is function code 0. Below is the slandered DNP3 data frame which was curled from (Clarke and Reynders, 2004 P.87).

Start	**Length**	**Control**	**Destination Address**	**Source Address**	**CRC**	**User Data**	**CRC**
2 bytes	1byte	1byte	2 bytes	2bytes	2bytes	16bytes	2bytes
0923	05	A0 (DIR=1, PRI=1, FCV=0, FC=RESET. From primary to secondary)	000B	0010	AB65		

0923	05	00 (DIR=0, PIR=0, FC=ACK. From secondary to primary)	0010	000B	CA53		

Table 4-5 Standard DNP3 Data Frame

Table 4.6 below shows the function codes that are sent from primary to secondary and from secondary to primary. Primary in this case is the initial message that was sent, or the original message and the secondary might be a response to the primary message. The figure 4.5 below will illustrate or show the message flow of the DNP3 showing the first byte of the message, which is the Application Control information (AC). However, the AC information is used to direct or control the message flow. This is made up of sequence number and the flags (East, Butts, Papa, and Shenoi, 2009).

Function Code	Description	Function code	Description
0	Confirm ACK-Acknowledging the receipt of the message	0	Reset Link (Remote link)
1	Confirm NACK-No acknowledgement due to busy. The message is not accepted.	1	Reset of user process
11	Response – Link Status. (DCF = 0 means free to send and 1 means not free)	2	Test link (Function for the link)
14	Link not working or functioning	3	Confirmation of user data is expected
15	Link service not in use or has not been implemented	4	No confirmation is expected of the user data.
		9	Request link status

Table 4-6 Secondary to Primary and Primary to Secondary

The communication flow between devices follows a normal handshake as in computer communication, see figure 4.5. Thus, the master sent a request to the slave or vice versa (1) and the request can be received or not received. When the request is received, an acknowledgement is sent to the sender, in this case it is a confirmation (2). The confirmation of a message received will depend on whether it was sent by the sender. Some messages are sent without setting the confirmation, thus the message will be received without acknowledgment. The number 3 is the response to the request by sending the requested information. However, when the information is received, the master responds with a confirmation of the receipt. If no confirmation is received, the slave will resent it again with the believe that the one sent previously was lost (Lu, Wang, and Ma, 2013).

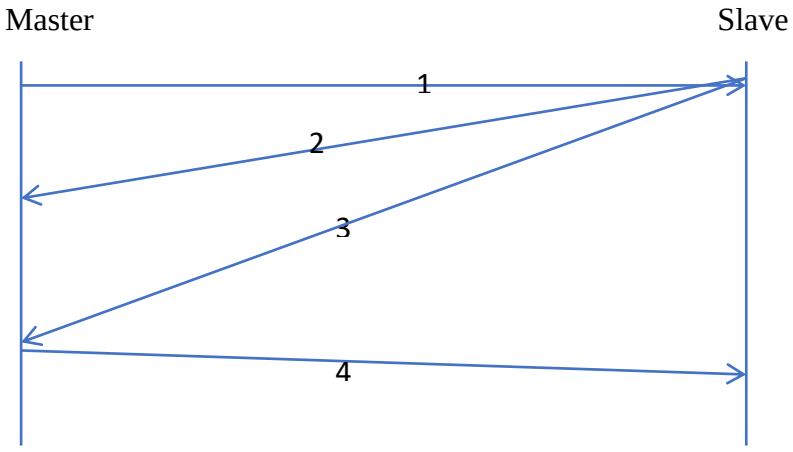


Figure 4-5 Communication flow of DNP3

4.2.3 ProfiNet Protocol

ProfiNet is an ethernet standard communication protocol that was developed by Profibus. Profibus on the other hand is a bi-directional communication protocol for some of the ICS devices. The ProfiNet is the current development that is faster and can accommodate more data on the system while Profibus still remain the classical model for serial fieldbus as explained by Verwer (2010). The ProfiNet protocol is fast and the transfer rate is in the range of 100 milliseconds for TPC/IP, maximum of 10 millisecond for Real-Time protocol and even less than 1 millisecond for Isochronous Real-Time protocol.

ProfiNet complies to the open standard industrial automation which is based on the ethernet communication. This is what differentiates it from the fieldbus systems. However, many automation product manufacturers used ProfiNet for their device communication and one of them is Siemens. Devices connected through ethernet forms 38% of the whole automation

devices and ProfiNet has the market share of 8% from this according to Automation World (2016). In 2017, the report from HMS showed that the ethernet communication accounts for 46% of the market share and ProfiNet was 11% from this 46% (Carlsson, 2017). This shows the important of this particular protocol in the automation world. Siemens S7 PLCs and HMIs uses this protocol for communication between its devices and others in the network. This protocol introduction will be narrowed down to only S7 devices from Siemens.

There are quite a number of S7 devices from Siemens such as S7-200, S7-300, S7-400, S7-1200 and S7-1500. These are devices that uses ProfiNet for their communication with other devices on the network. However, the S7 communication protocol are of two types for their identification or named protocol identifier which are; 0x32 and 0x72 as stated by Kleinmann and Wool (2014). The 0x32 types are the older model of the S7 devices such as S7-200 to S7-400, while the 0x72 is from S7-1200 (Verwer, 2010).

4.2.3.1 General Structure of the Protocol

The S7 protocol is mostly encapsulated in TCP and transported through port 102. The whole encapsulation involved the Header, or the telegram, ISO on TCP and the TCP/IP. The telegram is made up of the header, a set of parameters, parameter data and the data block. The ISO on TCP is made up of the TPKT which stands for Transport Packets and Connection Oriented Transport Protocol (COTP) and the S7 Protocol Data Unit (PDU). The last is the TCP/IP which is made up of the header and the ISO TCP telegram. The whole packet is encapsulated in TCP/IP for the transportation and communication with other devices on the network. Below in figure 4.6a is the image of the general structure of S7 protocol from Snap7 (2017).

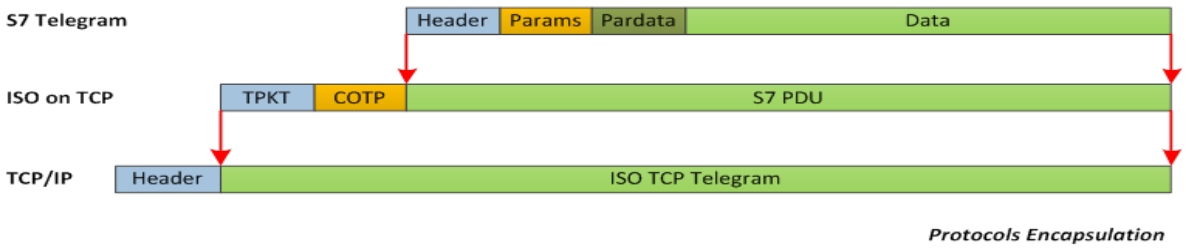


Figure 4-6a S7 Protocol Structure, Source Snap7 Sourceforge.net

The normal S7 message is structured as seen in figure 4.6a or in simpler form as seen in figure 4.6b

Protocol ID	Message Type	Reserved	PDU Reference	Parameter Length	Data Length	Error Class	Error Code

Figure 4.6b: Protocol Header

The protocol ID as earlier shown can either be 0x32 or 0x72 which contains 1 byte of data, while the message type represents the type of the message such as job request (0x01), acknowledgement(0x02), user data (0x07) which mainly contains one byte of data. The reserved are mainly 0s and the PDU reference is 2 bytes of data which is mainly being created by the master. Parameter and data length are 2 bytes of data while Error class and code are 1 bytes of data each. However, the PDU contains more data as seen in figure 4.6c (Miru, 2016)

Protocol Id	ROSCTR **(Message Type)**	Reserved
Request Id		Parameter Length
Data Length		Error Code **(for Remote Operation Service Control 3)**
Function Code	Item Count	

Figure 4.6c: S7 PDU

The function code here communicates the reason behind the message that was received or send such as read (0x04) and write (0x05). This can be a request to download or upload blocks to the system or from the system. Each of this type of communication has their own function code that determines the purpose. These communications are mainly between the master and the slave. The master initiates the communication with the slave and the slaves responds with the acknowledgement and acks accordingly (Verwer, 2010).

4.2.4 IEC 60870-5:

IEC 60870-5 is a standard produced by the International Electrotechnical Commission in 1988, Clarke and Reynders (2004 p.177). The part 5 is the transmission protocol, which has some documents that explained their function. This was released as section IEC 60870-5-1 to IEC60870-5-5. There's also a companion that defines the protocols at the application level. They define how the protocol works under the application. However, these companions are represented as IEC60870-5-101 to IEC60870-5-104 or sometimes represented as T101 to T104. The T here is referred to as Telecontrol for communication between devices. Below in figure 4.7 is a brief definition of the sections and the companions for transmission protocol for IEC60870-5

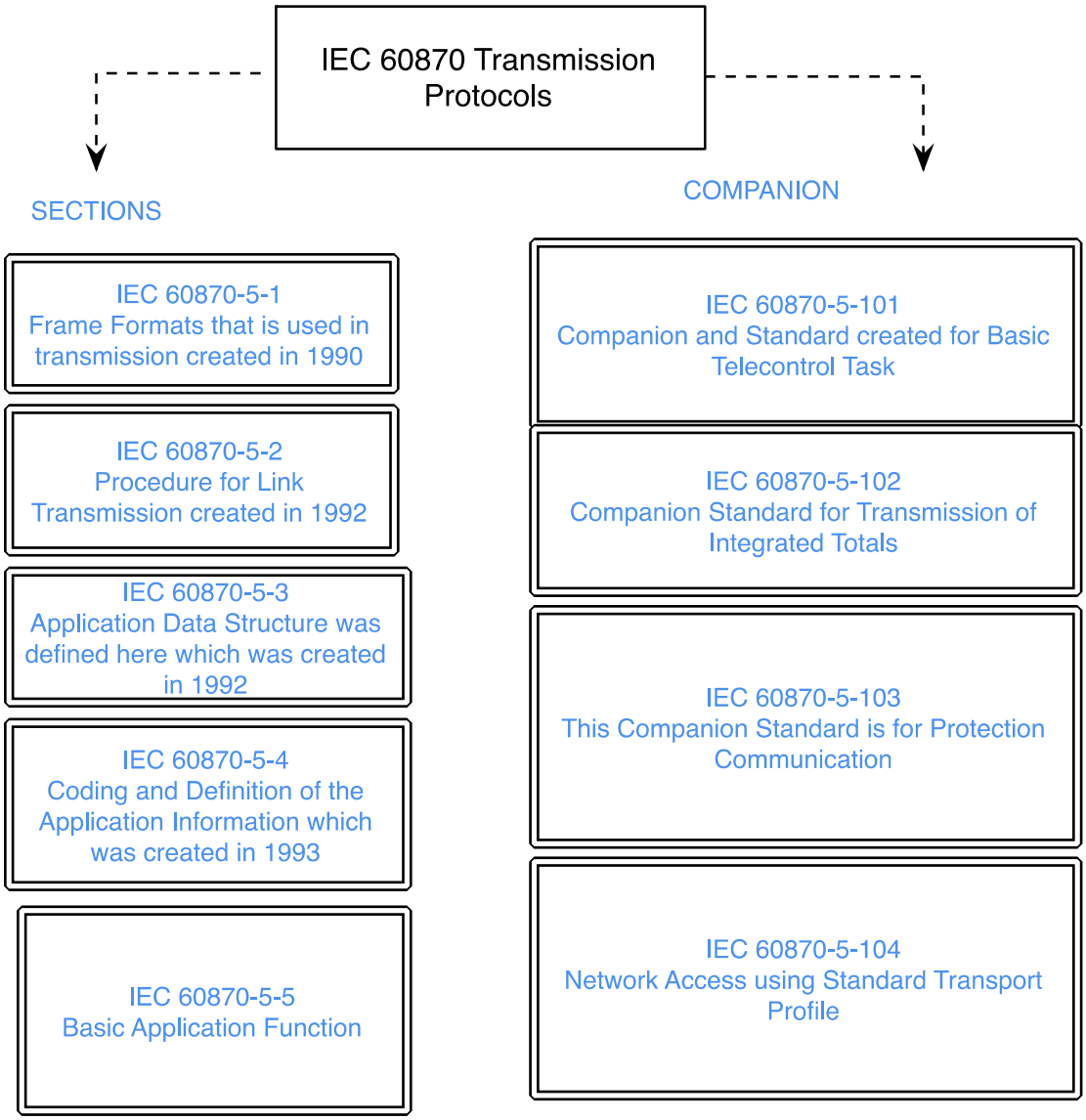


Figure 4-7 Sections and Companions for IEC 60870-5panions for IEC 60870-5

Figure 4.7 above is a brief description of the Section and companions in IEC 60870-5. It described the meaning of each section and companions with regards to the number that was attached to it. IEC 60870-5 is a three layers protocol for companion T101 with the following layers; physical layer, Data link and Application Layer with the addition of user area or user process. The T104 is a bit different because it includes the Network and Transport layers (Yang, McLaughlin, Littler, Sezer, Pranggono, and Wang, 2013).

4.3 Requirements

The proposed study is meant to detect both known and unknown anomalies on the ICS. Both the architecture and the framework will be a head start into modelling IDS by mimicking swarm of bird's movement and their detection strategy. The whole framework will help personnel in

virtualizing the system activities as well as equipping them with the necessary information for the security and reliability of the system.

Taking into consideration the Stuxnet (Langner, 2011) attack, it is necessary to design a system that will analyse every command against the system reliability, functionality and security. It is necessary that such a system would satisfy these requirements. The system should be able to resolve the issue of big data challenges the current IDS is now facing. This can be in form of speed of the data as well as the volume of data. The recent attacks on ICS as shown in chapter 3 caused physical damage to the systems. However, the only way to minimise this is to design an architecture that will ensure reliability as well as functionality of the system. in order to ensure that these functions are included in the system, data from the system will be collected.

Data will be generated from the SCADA system or testbed that will be used for this project. It was deemed necessary to generate the dataset as the purpose is trying to detect when an issue will cause the system to be unreliable and malfunction. For instance, a Train system can be manipulated to purposely increase speed or not to stop where it supposed to, which might cause the train to collide with another Train. However, many of these and others will be performed in the lab for testing and analysis purposes.

4.3.1 Functional Requirements

Functional requirements are the things that are necessary or must be integrated into the system in order to make it acceptable by the users (Whitten, Bentley and Dittman, 2001, P. 215). These requirements must be written using action verbs or words. This can start with the objectives of this project and the hypothesis. By meeting the hypothesis and the objectives, the functional requirements are met. Below are the functional requirements that have been discovered till now for this project.

- Train system that will use the SCADA field devices as a testbed will be built
- Calculate the performance of the system based on the data collected
- Integrate the system to the cloud
- Model the system and birds that will detect anomalies
- The system should detect both known and unknown anomalies

4.3.2 None Functional Requirements

None functional requirements are the description of the proposed features as well as constraints (Whitten, Bentley and Dittman, 2001, P. 216). This will show the limitation that one will encounter in the project. None functional requirements will state the acceptable performance as well as the detection and other information that is needed for the system performance.

- The latency should not be more than 200 milliseconds for a better performance
- The data from the SCADA should be stored in the cloud
- The SCADA system will interface with the cloud
- The protocol should be used for detection
- The security should be based on functionality and reliability.
- The algorithm should be distributed in nature
- A scalable database should be used that can accommodate both structured and unstructured data.

Table 4.7 is the requirement for the selection of the database based on the aim and objectives of this project. Informed research and decision were taken, and consideration was made based on the aim of this project. The SQL and NoSQL databases were considered based on their suitability and how they can add value to the project. It is obvious the reason HBase was selected as the database as seen in table 4.7 for this project. HBase as a NoSQL database accommodates both structured and unstructured data and it is simple and easy to use. SCADA data can be both structured and unstructured depending on the environment at which the system is operating. NoSQL as defined by Leavitt (2010) means non-SQL or non-relational data base. The relational database such a MySQL and oracle are based on tables or tabular relation which

Database	Scalability	Distributed	Structured/ Semi-structured and simple	Maintenance/ Simplicity
MongoDB	✓	✓		
RDBMS				
Cassandra	✓	✓		
Hbase	✓	✓	✓	✓
HCatalog	✓	✓		✓

Table 4-7 Databases for Big Data

cannot handle semi-structured data as well as other data that fall outside the tabular form. This is one of the major reasons for NoSQL as well as its scalability and simplicity in design and operation. HBase works well with apache Hadoop and MapReduce in handling Realtime data. Compared to other databases, it scales horizontally and can handle billions of columns and rows of data. Without further elaboration on other databases, table 4.7 summarises the reason for HBase amongst other databases.

Database might not be able to function on its own, therefore there are other technologies that will allow the implementation of the database and the system. Chosen the right technology for the system will depend on certain criterial as seen in table 4.8. Hadoop framework and cloud computing were chosen as seen in table 4.8 because of the attributes stated there and more. However, the compatibility between the two technologies made it easy to provide a computer service at a very cheap rate and easy to maintain if well implemented. With the two together, complex algorithms can run on the system very fast and efficient (Yamijala, 2013).

Technology	Distributed	Simplicity	Scalability	Cheap
High Performance Computer	✓			
Hadoop	✓	✓	✓	✓
Distributed Computer	✓			
Cloud Computer	✓	✓	✓	✓

Table 4-8 Technology

4.4 System Architecture

The proposed architecture below in figure 4.8 is made up of four parts. The first part is the data source part, which are as follows; sensors, valves and actuators as well as data from the network. The second part will be called the communication part or the field devices, which is made up of the Programable Logic Controller (PLC) and Master Terminal Unit (MTU). The third part is the ALF where the algorithm resides, and the fourth part is the cloud with Hadoop framework for data storage and processing.

As packets are being received from sensors, actuators and valves by the PLC and the MTU, it is being compared by the Artificial Life Framework (ALF). The comparison of the data is to

avoid the possibilities of data being corrupted on the flow or being altered. The algorithm will compare and analyse incoming data for any anomalies. Data will further be processed in the cloud for storage. Apache spark is used for data streaming and Flume will be configures for data aggregation or own Ingester will be written for ingesting data into the cloud system.

Another aspect of the bird model or the Artificial Life Framework (ALF) is that it will also compare and corroborate with other available information in order to help in making an informed decision. This project will use train system as a case study for SCADA system. Trains run on the tracks and controlling the flow can be very difficult at times. Therefore, some detection model is needed that can check and corroborate with other stations in order to make an inform decision that will avoid collision. Birds corroborate with each other in the flock as they move, the same way information can be transferred from one station to another in order to guide the bird's decision. This will keep this system reliable and functioning at all time.

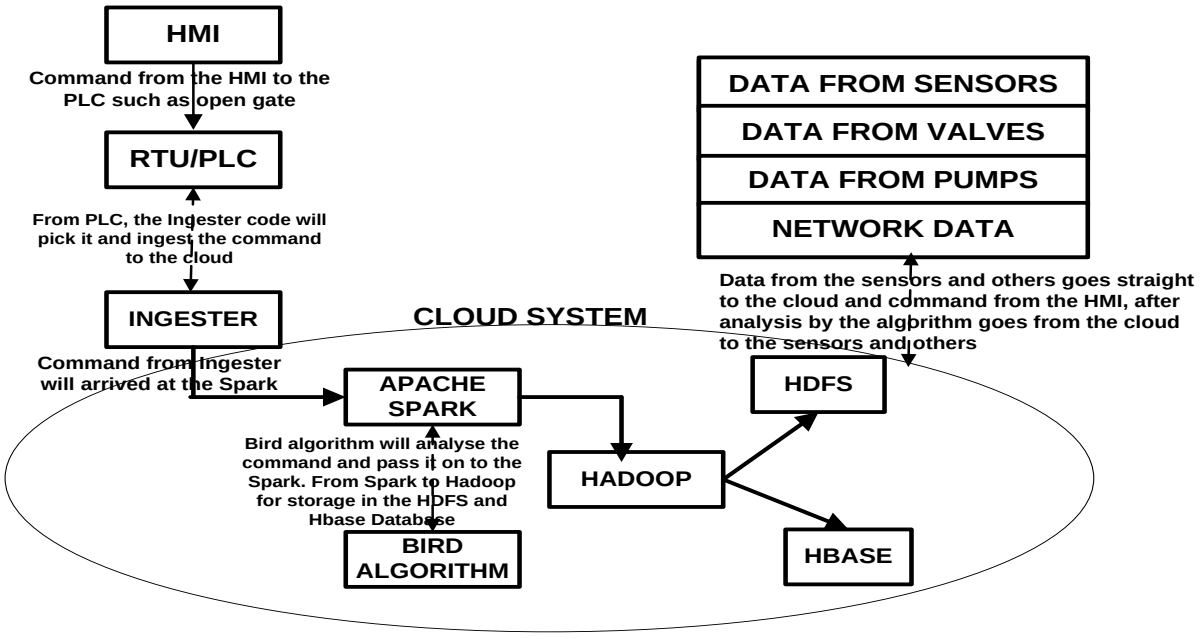


Figure 4-8 System Architecture

4.4.1 Designing a Case Study

Supervisory Control And Data Acquisition (SCADA) will be used as an environment for the case study. This will be developed for the initial data collection and to support the integration and testing of the detection of anomalies in the system. The setup of the environment and its constituents will be as seen in figure 4.9. Figure 4.9 is the logical design of the case study which

will be used for data collection. This will be the test environment which consists of a number of variables as seen in section 4.4.2 and 4.4.3 below.

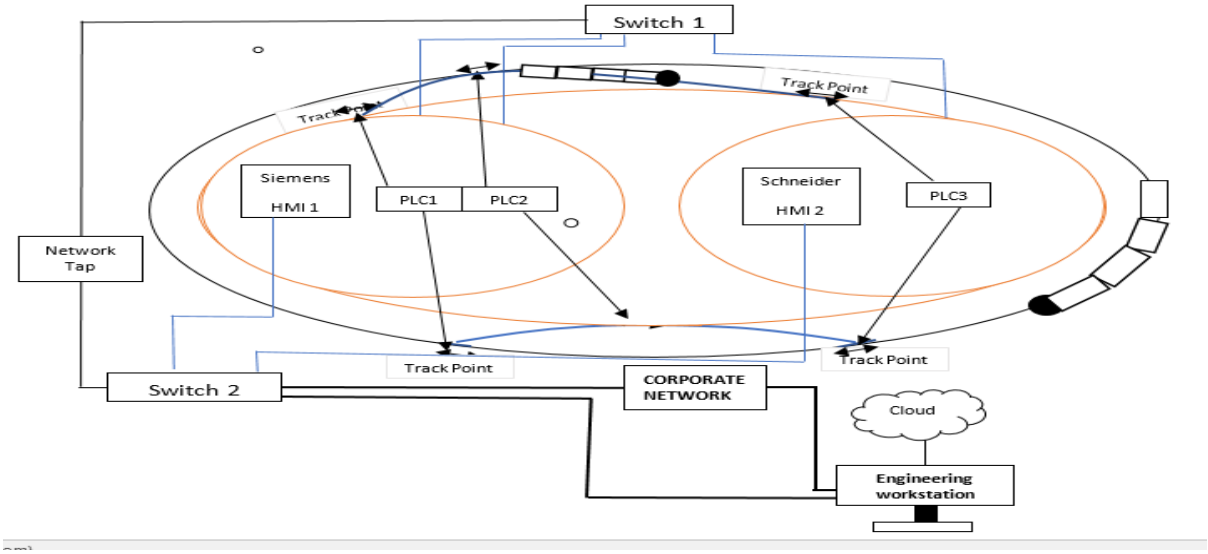


Figure 4-9 Logical Diagram of Case Study Environment

4.4.2 Variables

- Rail track (with two oval routes connecting via two turnouts)
- 2 x Arriva Train Wales Locomotives + 2 x Arriva Train Wales passenger carriages

4.4.3 Control Hardware

- 2 x Siemens S7 1212c Programmable Logic Controllers
- 1 x Siemens HMI
- 1 x Schneider Electric Modicon PLC M221 TM221CE16R
- 1 x Schneider Electric HMI
- Engineering workstation running Siemens Step 7 and Schneider electric SoMachine Software
- 2 x Switches TP LINK TL SG 1008 and TP LINK TL SG 1024
- 10 Hornby Point Motor R8014 and their covers R8015 for track change
- Hornby DCC Controller (for directional and speed control of locomotives)

Below in figure 4.10 is the physical view of the real train system that is equipped with the PLCs, HMIs and sensors. The system is in the lab and has all the SCADA devices and software that will help in collecting data for comparison and other analysis.

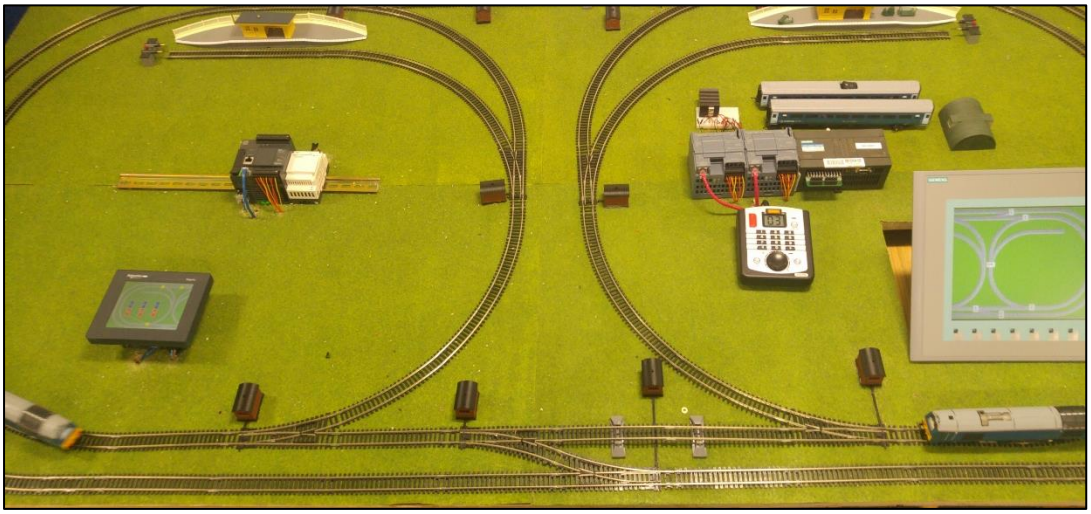


Figure 4-10 The Train System

4.4.4 Operation

Siemens HMI will be responsible for communication, control and issuing of commands to Siemens PLC1 and PLC2. The two PLCs will be responsible for the operation of the level crossing barriers, track points for the route change and signal light. The engineering workstation will monitor memory changes to both HMIs and PLCs. Schneider HMI will be responsible for Controlling the Schneider PLC. The PLC will receive its command from the HMI such as open track point, close track point for the change of lane or route

4.4.5 Purpose of this Test Bed

Simulate a realistic SCADA based environment

- Collect data that will help for the testing and analysis of our model
- For the proper modelling of the communication
- Virtualisation of all possible attack scenarios for the purpose of modelling
- Tool testing against SCADA environment for collection, preservation and reporting
- Low level data extraction techniques for embedded devices

Below in figure 4.11 is the testbed that shows how the data will be collected from the system. The logical view of the testbed in figure 4.11 below is a reflection of physical system in figure 4.10. However, the logical view integrated the data collection point as well as how the PLCs and HMIs were arranged in the system. Data will be collected using either the engineering

workstation or the computer with access to the cloud computing. This computer is a laptop that was connected to the system through the network switch.

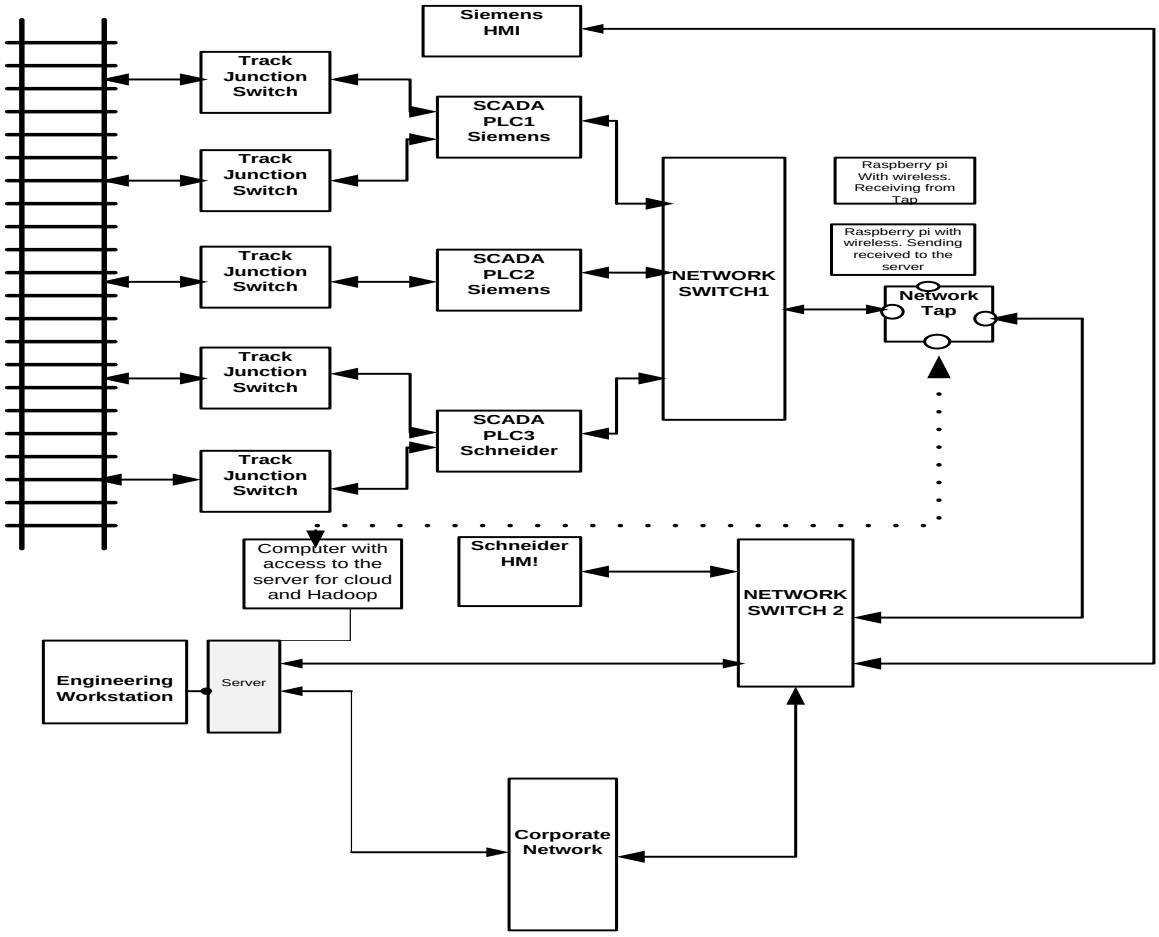


Figure 4-11 Testbed for the Model

4.4.6 Reason for this Setup

This study acknowledged that there are quite a number of emulation software available that can be used in construction of SCADA system. This study experiments with some of these software applications such as Wonderware, Schurter, RapidScada and amongst others. These applications can be used to design and experiment as well as simulate a real SCADA environment. Many studies and scientific research have followed that procedure in obtaining some results.

According to Wood and Goldenberg (2013), due to the sensitivity involved in the SCADA environment, real SCADA data is difficult to get. This is due to the nature of SCADA system. The operation is meant not to be interrupted as it runs for 24 hours round the clock. Although simulation can produce data of the same kind, but there might be some variations. In a real environment, you will be able to monitor the actions and functionalities of the components of

the system. The reason for using simulation is the lack of the real environment for the experiment. Therefore, this research and experiment will be carried out on a real components of SCADA system. Although the simulation will yield robust system, but the real environment will yield real data and correct findings.

4.5 Data Collection and Analysis

The methodology, see appendix, outline the methods and approaches that will be used in achieving the aim of this research. The Design Science methodology was chosen which involved the creation of artifact. Hence the artifact can be in form or a model, construct, design theory and among others. This research will produce artifacts inform of models that will satisfy the objectives of this research project. However, case study and experimental approaches has been chosen for data collection and experimental purposes. The Train system or locomotive system represents the ICS environment, which will be used as a testbed for the model and for data collection purposes.

This section of the chapter will deal on data collection and analysis of the collected data. It will define what data should be collected, where and how the data will be collected from the system as well as the type of data. There are several types of data in SCADA system, therefore data collection will specify the type needed for this research. The type of data will be known from the initial data collection approach from all the PLCs and HMIs in the system. As stated by Yin (2009), so many people chose case study approach based on their initial projection of the outcome of the research. Case study data collection without a proper plan will always lead to failure. Patil and Yogi (2011) emphasised on the project failures due to improper data collection. They emphasised on the tendency of the researchers to involve in guessing the data at some point. There are many things that can go wrong during the data collection but most of them have been taken care of such as;

1. Absence of a proper system where the data could be collected
2. Time constraints for the data collection. This can allow guessing the outcome of the data due to limited time.
3. Out of date data which can cause inaccuracy of the data. For instance, there are SCADA protocols online from Siemens PLC. Most of this data are from Siemens S7 200, 300 and 400 series PLC.

These issues will be addressed by making out time for a meticulous examination and collection of data. This will show the data types from all the PLCs and the HMIs inform of the function blocks and tables. The collection strategy for this data from the PLCs and HMIs will be devised and then the analysis of the collected data to help in making an informed decision with regards to latency, data type and state of the sensors in the system.

4.5.1 Types of Data

The Train set or the locomotive system that was used in the project has two types of PLCs namely; Siemens S7 1212c and Schneider Electric PLC (Modicon) M221 TM221CE16R. They are programmed with the instructions for the Train systems such as when to open the sensors for the tracks and for the change of lane. This information is programmed using Totally Integrated Automation (TIA) portal for the Siemens devices and SoMachine for the Schneider Electric devices. Below are the data types that can be used in the PLCs for the space in the memory and their sizes (Siemens S7 1200, 2015).

Data Type	Bit Size	Numerical Range	Constant Entry Examples
Bool	1	0 to 1	TRUE, FALSE, 0,1
Byte	8	16#00 to 16#FF	16#12, 16#AB
Word	16	16#0000 to 16#FFFF	16#ABCD, 16#0001
DWord	32	16#00000000 to 16#FFFFFFFF	16#02468ACE
Char	8	16#00 to 16#FF	'A‘, 't‘, '@‘, ‘∑’
Sint	8	-128 to 127	+50, 16#50
Int	16	-32768 to 32767	30000, +30000
Dint	32	-2.147.483.648 to 2.147.483.647	-2131754992
USInt	8	0 to 255	78, 2#01001110
UInt	16	0 to 65.535	65295,0
UDInt	32	0 to 4.294.967.295	4042322160
Real	32	+/-1.18 x 10-38 to +/-3.40 x 10 38	123,456, -3.4, -1.2E+12, 3.4E3
LReal	64	+/-2.23 x 10-308 to +/-1.79 x 10 308	12345.123456789 -1.2E+40
Time	32	T#-24d_20h_31m_23s_648ms_ to T#-24d_20h_31m_23s_647ms Stored as -2.147.483.648 ms to +2.147.483.647 ms	T#5m_30s 5#-2d T#1d_2h_15m_30x_45ms
Time_of_Day	32 bits	TOD#0:0:0.0 to TOD~23:59:59.999	TOD#10:20:30.400 TIME_OF_DAY#10:20:30.400 23:10:1
String	Variable	N = (254 byte)	'ABC ‘

Table 4-9 Data Types Table

The data type for both the Siemens and the Schneider devices is Boolean data type. The sensors are going to be programmed for ON and OFF. Switching it for one direction is turning it on for that particular direction and the other direction will be in the OFF state. According to table 4.9, the Boolean is a state of true or false. When the state is ON, the condition is true and when in OFF state, the condition is false. Table 4.10 shows the data type of the systems program, configurations and the digital output addresses. Table 4.11 shows the Boolean data type from Siemens PLC while Table 4.12 is from Schneider Electric PLC.

PLC tags

	Name	Tag table	Data type	Address	Retain	Visibl...	Acces...
	Switch A	Default tag table	Bool	%M0.0		☑	☑
	Switch A left	Default tag table	Bool	%Q0.0		☑	☑
	Switch A right	Default tag table	Bool	%Q0.1		☑	☑
	Switch B left	Default tag table	Bool	%Q0.2		☑	☑
	Switch B right	Default tag table	Bool	%Q0.3		☑	☑
	Switch B	Default tag table	Bool	%M0.1		☑	☑
	Switch C left	Default tag table	Bool	%Q0.4		☑	☑
	Switch C right	Default tag table	Bool	%Q0.5		☑	☑
	Switch C	Default tag table	Bool	%M0.2		☑	☑

Table 4-10 Siemens S7 1200 PLC 1

PLC tags

	Name	Tag table	Data type	Address	Retain	Visibl...	Acces...
	Switch D left	Default tag table	Bool	%Q0.0		☑	☑
	Switch D right	Default tag table	Bool	%Q0.1		☑	☑
	Switch D	Default tag table	Bool	%M0.3		☑	☑
	Switch E	Default tag table	Bool	%M0.4		☑	☑
	Switch E left	Default tag table	Bool	%Q0.2		☑	☑
	Switch E right	Default tag table	Bool	%Q0.3		☑	☑
	Switch F	Default tag table	Bool	%M0.5		☑	☑
	Switch F left	Default tag table	Bool	%Q0.4		☑	☑
	Switch F right	Default tag table	Bool	%Q0.5		☑	☑
	<Add new>					☑	☑

Table 4-11 Siemens S7 1200 PLC 2

Symbol list

Address: Symbol:

Used	Address	Symbol
☑	%M0	HMI_0
☑	%M1	HMI_1
☑	%M2	HMI_2
☑	%M3	HMI_3
☑	%M4	HMI_4
☑	%M5	HMI_5
☑	%Q0.0	OUT_0
☑	%Q0.1	OUT_1
☑	%Q0.2	OUT_2

Table 4-12 Schneider PLC M221 TM221CE16R

Both the input and the output are programmed into the block diagram of the SoMachine from Schneider Electric and the TIA portal or Ladder Logic from Siemens. The program block in the PLC will be virtualize in the HMI for the operational purposes. Below in figure 4.12 to 4.14 are some of the block programs that were used for the testbed. Figure 4.12 and 4.13 are the block program for the Siemens TIA. These are the program that will direct the operation of the devices in the environment such as open and close the sensors. These sensors are connected to the PLC and they receive information directly from the PLC. Each of them has their addresses

and through their addresses, their state can be known, whether ON or OFF. In figure 4.14 is the block diagram or program for the Schneider Electric PLC. This information was programmed in the SoMachine for the Schneider Electric PLC and HMI. The outlook is quite different compared to that of Siemens TIA. It has the address, input and the output which is simpler in the appearance compared to other manufacturers.

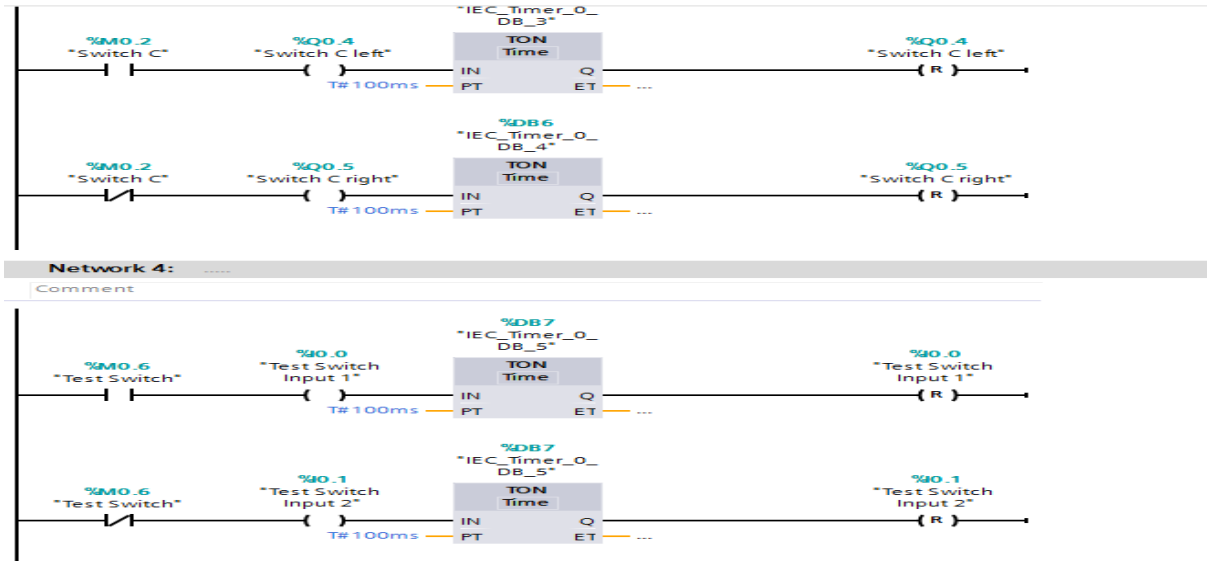


Figure 4-12 Siemens PLC 1 blocks program for the point motor

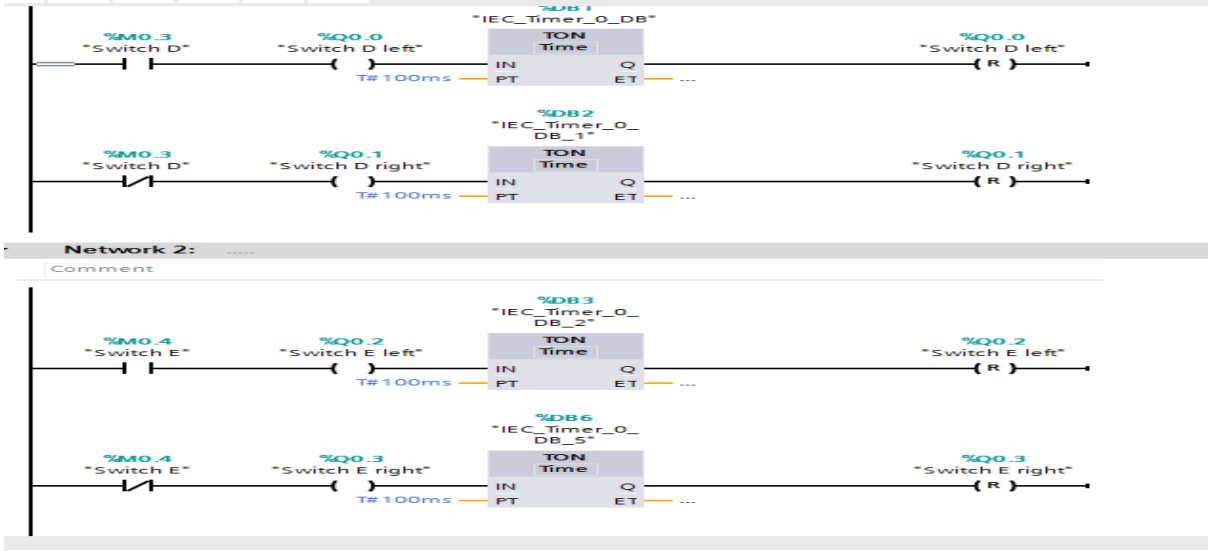


Figure 4-13 Siemens PLC2 blocks program for the point motors

Figure 4-14 Schneider PLC blocks program for the point motors

4.5.2 Data Collection Strategy

The data from the PLC will be collected by configuring a data logging for the web server on the PLC. This data will be used to measure the open and closed point motors (sensor). With this data, it would be easier to know the sensors or tracks that might course problem at any particular point in time. Another data that will be collected is the data from the HMI. This will help in determining the location of the locomotive at a particular time. Below in figure 4.15 is the data collection approach and the reason for collecting such data.

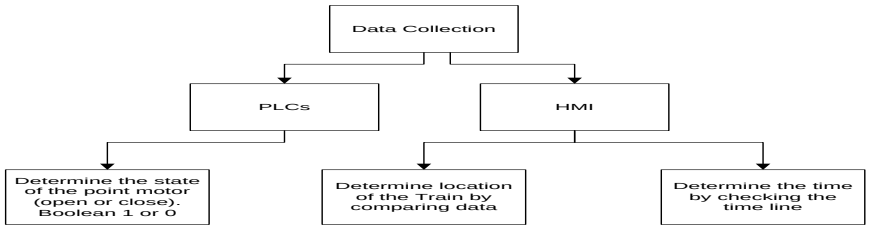


Figure 4-15 Data Collection strategy

4.5.2.1 Siemens PLC S7-1212c Processor

The data gathered from the PLC are the data that will help in determining the state of the sensors. Table 4.10 and 4.11 showed that the data type in the program is the bit data type or Boolean represented as Bool in the table. This shows the type that is expected to be collected from the PLC. However, there are other data types as stated in table 4.9, but at the moment, this is outside the scope for the data collection stage. The reason being that, the sensors used for the test bed are either true or false. It is either open or close and the open means that the locomotive or Train can enter. The two Siemens PLCs have each three blocks of program they are controlling, because the sensors are wired directly to the PLCs input and output. Below are the data that shows the state of the sensors based on the block program above in figures 4.12 to 4.13.

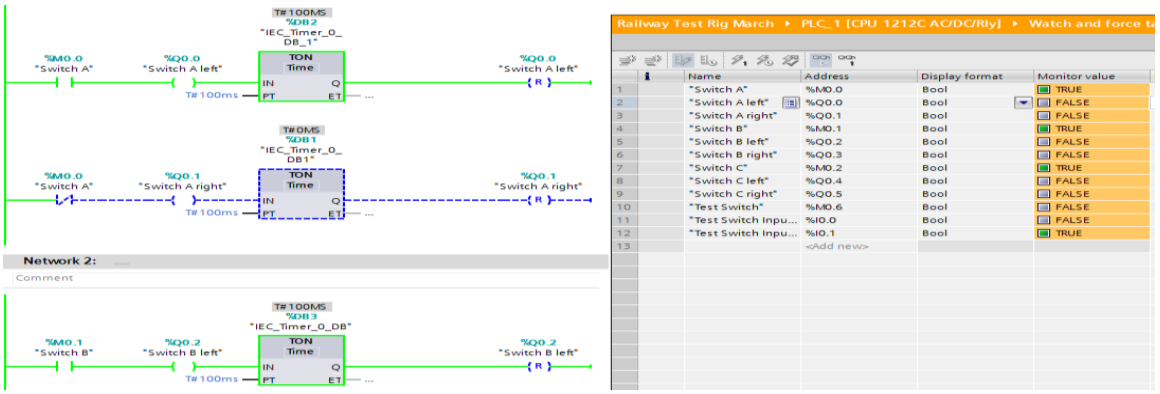


Figure 4-16 Data from the Siemens PLC

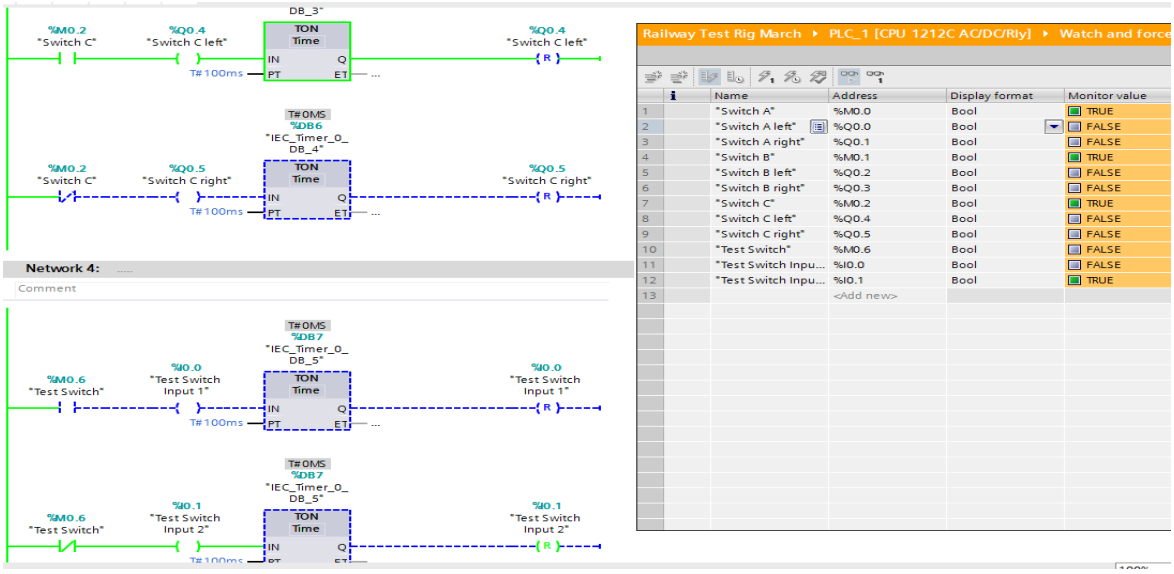


Figure 4-17 Data from the Siemens PLC

In figure 4.16 and 4.17, the information on the left and right screen showed the states of inputs; M0.0, M0.1, and M0.2 are True. On the left side of the same figures, the first block is green which shows that it has a state of 1 or True. The block is normally open block which is activated when the button in the HMI is pressed. For the experimental purposes, figure 4.17 showed that M0.6 is False. This showed that the Train cannot enter the direction as the direction is not open for train entrance.

However, the information in the TIA portal has been collected for the Siemens S7 PLCs. This information will be used to determine the state of each sensor on the rail; whether they are open or close. This will help in avoiding collision because the risk scenarios will be mapped out. The collected data will help in mapping out the scenarios that will cause collision, whether

intentionally activated or unintentionally performed. From the collected data, there are 7 vital networks based on figure 4.16 and 4.17. One of them, the M0.6 was there for the purpose of this experiment in order to show the state of the input. The M0.0 to M0.5 addresses are all in the True state, which means that they are all open. These are the six areas or addresses of the Siemens PLCs that will help in determining the risk condition or areas for a situational awareness of the operator. Situational awareness is to inform an operator of the state of the systems. This can be pictorial or alert, depending on the configuration.

PLC 1 = M0.0, M0.1 and M0.2

PLC2 = M0.3, M0.4 and M0.5

In order to measure the risks associated with the two Siemens PLCs. There are things to determine, the direction of the locomotive is very important. The scenario will be for only one direction of the locomotive. The question now is the likelihood of collision with two or more locomotives on the track based on the 6 input points? This can be determined by further analysis during the case study that will be carried out later.

4.5.2.2 Schneider Electric PLC

The function blocks in the Schneider electric PLC TM221CE16R was programmed with the SoMachine Basic version 1.4. With the software, it was hard to retrieve all the required information needed. This posed as a constraint for information retrieval in the system. Table 4.13 below is the digital output from the Schneider software called SoMachine which was used for the PLC programming.

tal outputs

Used	Address	Symbol	Used by	Status Alarm	Fallback value
☑	%Q0.0	OUT_0	User logic	☐	0
☑	%Q0.1	OUT_1	User logic	☐	0
☑	%Q0.2	OUT_2	User logic	☐	0
☑	%Q0.3	OUT_3	User logic	☐	0
☑	%Q0.4	OUT_4	User logic	☐	0
☑	%Q0.5	OUT_5	User logic	☐	0

Table 4-13 Digital Outputs from the Schneider Electric SoMachine

The information in the digital output is the output addresses as it was stated in the block diagram as well as the symbols. This information was captured during the simulation of the system. The system has been programmed to send short electric pulse to the sensors for the

opening of the track points (sensor) for the locomotive or Train to change direction. However, the output symbols in table 4.14 are six in number, but they control three sensors in the system as seen in table 4.14 below. The reason is that each sensor has left and right switch. Each direction sends electric pulse when pressed, therefore each condition or state of the sensor is recognised.

Table 4-14 Schneider Electric PLC Sensors Locations and their values

In table 4.14, some addresses were set to 1 in order to show that it is possible to change the state. The reason for allowing this to send electric pulse and continue in the state is for the PLC not to get too hot due to the activities. It shows that the state of the sensor can actually be determined by the value. It also showed that the information is from the PLC and the address starting from Q0.0 to Q0.5 representing six addresses in total. These addresses are places in memory of the PLC and to know the location, it would be easier to retrieve the information for comparison purposes.

Table 4.15 showed both the output or coils name and the addresses in the memory as it were on the function block. The question now is how to retrieve these data from the memory. Figure 4.18 will give the overview of the process of the input and output data in the PLC memory for both Siemens and Schneider.

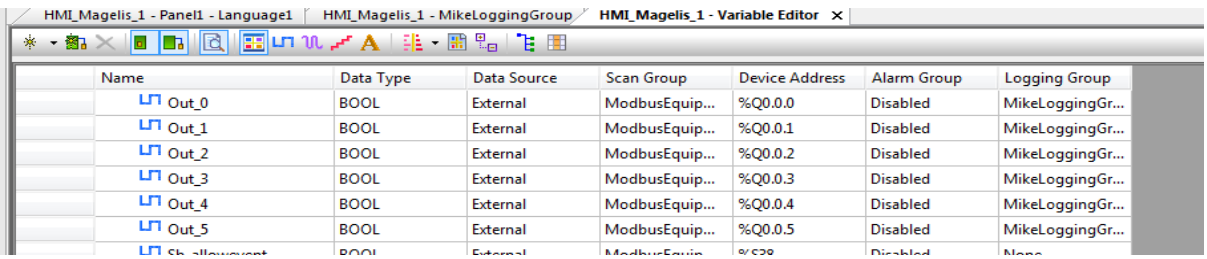


Table 4-15 Datalogging of the Schneider PLC

4.5.3 Bringing the PLCs Data Together

The block diagram that is seen in the TIA portal and the SoMachine are loaded to the PLC using the same program. This data stored in the PLC memory and the functions the data supported are recorded in the memory. The CPU validates the data at all time, both the input and the output data in the memory. Whenever a request is made as an input, the information is processed by the processor and the output will be the open or close of the track as seen in figures 4.16 and 4.17.

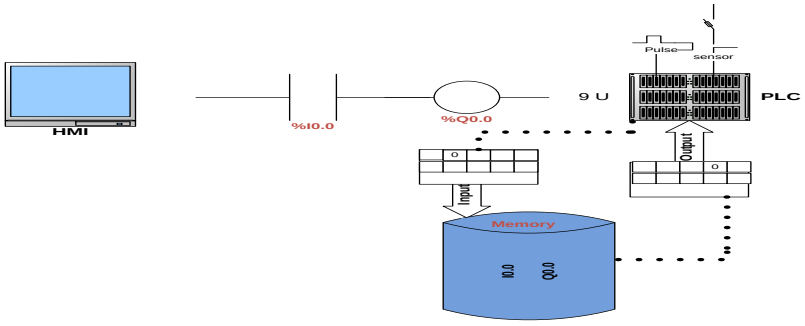


Figure 4-18 Data in the PLC Memory

The data from the PLCs, both the Siemens and the Schneider Electric were to determine the state of the sensors. When the risk conditions are determined, this will help in designing the systems for collision mitigation and avoidance. Before plotting the truth table of the risk's conditions and points, the TCP data will be collected for both Siemens PLCs and Schneider Electric PLC.

Protocol	Percent Packets	Packets	Percent Bytes	Bytes	Bits/s	End Packets	End Bytes	End Bits/s
Frame	100.0	1919	100.0	235866	2634	0	0	0
Ethernet	100.0	1919	11.4	26866	300	0	0	0
Internet Protocol Version 4	100.0	1919	16.3	38380	428	0	0	0
Transmission Control Protocol	100.0	1919	71.7	169114	1888	251	5020	56
TPKT - ISO on TCP - RFC1006	86.9	1668	2.8	6672	74	0	0	0
ISO 8073/X.224 COTP Connection-Oriented Transport Protocol	86.9	1668	2.1	5004	55	0	0	0
Data	86.9	1668	50.5	119058	1329	1668	119058	1329

Table 4-16 Siemens S7 1212c Protocol Packet Hierarchy

Protocol	Percent Packets	Packets	Percent Bytes	Bytes	Bits/s	End Packets	End Bytes	End Bits/s
Frame	100.0	880	100.0	58080	765	0	0	0
Ethernet	100.0	880	21.2	12320	162	0	0	0
Internet Protocol Version 4	100.0	880	30.3	17600	231	0	0	0
Transmission Control Protocol	100.0	880	48.5	28160	371	0	0	0
Modbus/TCP	100.0	880	18.2	10560	139	0	0	0
Modbus	100.0	880	7.6	4400	57	880	4400	57

Table 4-17 Schneider Electric M221 Protocol Packet Hierarchy

Table 4.16 and 4.17 are the packet hierarchy for bot Siemens S7 1212c and Schneider Electric PLCs. Table 4.16 is the data from Siemens device while table 4.17 is the data from Schneider Electric device that uses Modbus protocol. The information shows how the packets are arranged before sending it out to the receiver. This will help in finding exactly what information is needed for the detection purposes. Hence this is the reason for dissecting the protocols for more information on how it is being arranged.

From table 4.18 and 4.19, Schneider Electric PLC and HMI uses Modbus protocol and the structure of the protocol in figure 4.19 showed that Modbus is a simple protocol compared to Siemens in table 4.18. however, the most important thing is the required information that is needed in the packet.

```
0000  28 63 36 18 05 1d 28 63  36 80 f0 dc 08 00 45 00   (c6...(c 6.....E.
0010  00 79 e0 39 00 00 1e 06  a5 21 0a 14 01 34 0a 14   .y.9.... .!...4..
0020  01 c9 00 66 c0 0e 00 69  dd e1 00 6b 38 7e 50 18   ...f...i ...k8~P.
0030  20 00 58 23 00 00 03 00  00 51 02 f0 80 72 02 00    .X#.... .Q...r..
0040  42 32 00 00 05 4c 00 00  c0 c9 34 00 01 80 01 00   B2...L.. ..4.....
0050  02 80 01 01 03 80 01 01  04 80 01 01 00 00 81 ca   ........ ........
0060  80 3a 20 36 b5 34 43 9d  80 59 26 a9 1c c0 4f 32   .: 6.4C. .Y&...O2
0070  7d 09 80 86 42 23 84 33  20 c4 5c da 08 32 e6 74   }...B#.3  .\..2.t
0080  d2 eb 40 72 02 00 00                              ..@r...
```

Table 4-18 Siemens S7 1212c Message Structure

```
0000  00 01 23 2b e9 c8 00 80  f4 0d e3 0e 08 00 45 00   ..#+.... ......E.
0010  00 34 f7 32 00 00 40 06  6c 6d 0a 14 01 35 0a 14   .4.2..@. lm...5..
0020  01 c8 01 f6 2c 4c e2 90  00 61 9e 63 9f 56 50 18   ....,L.. .a.c.VP.
0030  11 1c 39 77 00 00 00 00  00 00 00 06 ff 0f 00 04   ..9w.... ........
0040  00 01                                             ..
```

Table 4-19 Schneider Electric M221 Protocol Message Structure

Figure 4.19 is the dissected Siemens S7 1212c data that was collected. The reason for this approach is to determine the data that will be needed and their location. The data here was collected and dissected using Wireshark tool (Wireshark, 2018)

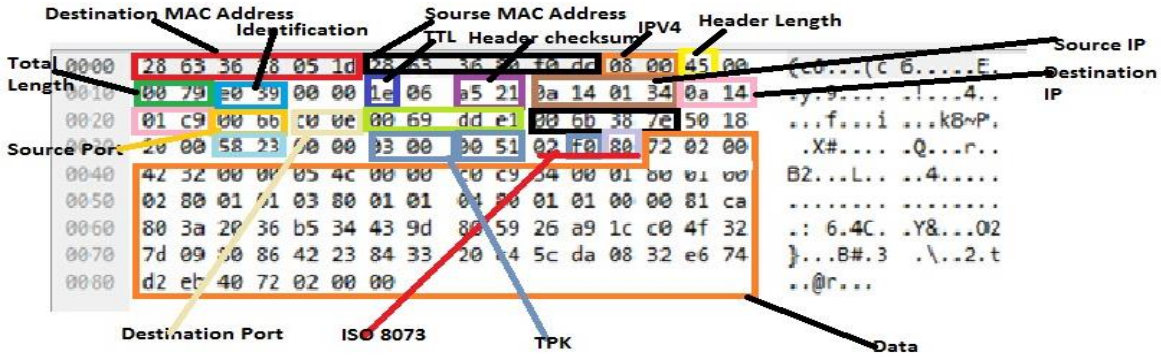


Figure 4-19 Siemens S 7 1212c Message Structure interpretation

From the structure of both protocols in figure 4.19 and 4.20, the information clearly showed that Modbus protocol is a simple protocol compared to ProfiNet. ProfiNet has some other layers that it is encapsulated with as seen in figure 4.19 above. The Modbus only has the TCP/IP and other protocol information as seen in figure 4.20.

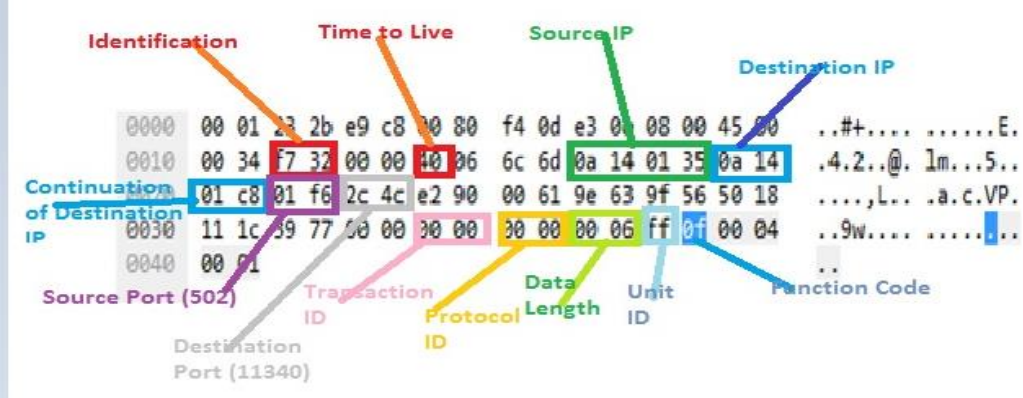


Figure 4-20 Schneider Electric M221 Message Structure Interpretation

The information that is needed both from Siemens device and Schneider as shown in figure 4.19 and 4.20 will determine how the parser will be programmed. When this information is captured by the parser, it will be passed further, for the detection purposes. This will help in the detection of anomalies in the system.

Figure 4.21 and 4.22 are the demonstration of packet flow in the system. The information showed some points where the number of packets received are higher. The points can be seen in both graphs and these were the times the buttons in the HMI for the sensors were pressed and continue to be pressed. As the button is pressed in the HMI, information is sent to the PLC and the response is the True or false information seen in figures 4.16 and 4.17 as well as in table 4.15 as discussed earlier. The purpose was to know the possibility of determining the position of each sensor with the TCP data. The information gathered so far has shown that

knowing the state of the sensor is possible. Although the sensors have only two possible states which can either be true or false or in ON or OFF states.

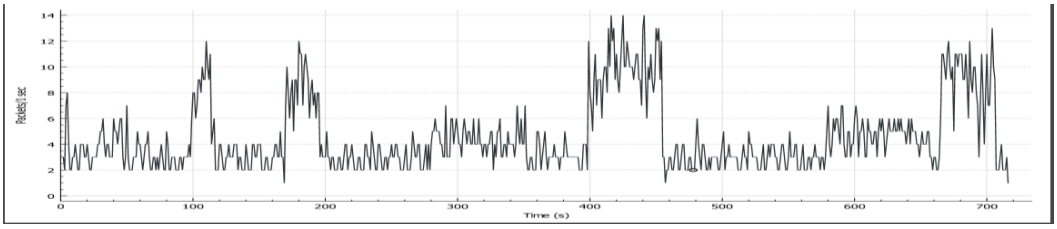


Figure 4-21 Siemens (ProfiNet) Packets Flow

The information in figure 4.21 also clearly showed the number of packets per second that flows in to the system. The highest number here was 14 packets per seconds which was between time of 400 and 460 seconds. With this information, it is possible to project the number of packets that is needed in the system per second depending on the capacity of the system. In order word, with the information, the capacity of the system can be measured. Although this cannot not be the maximum capacity of the system but with further collection and analysis, it will be determined. Figures 4.23 and 4.24 shows the sequence of the data flow between devices. The information in these diagrams is to help in determining the complete handshake between two devices. Sequence of data flow into a system is very important because it will also help in identifying anomalies in the system. If the data received is in the same sequence as it was sent, there will be no problem. Data can be lost and received in different sequence as it was sent.

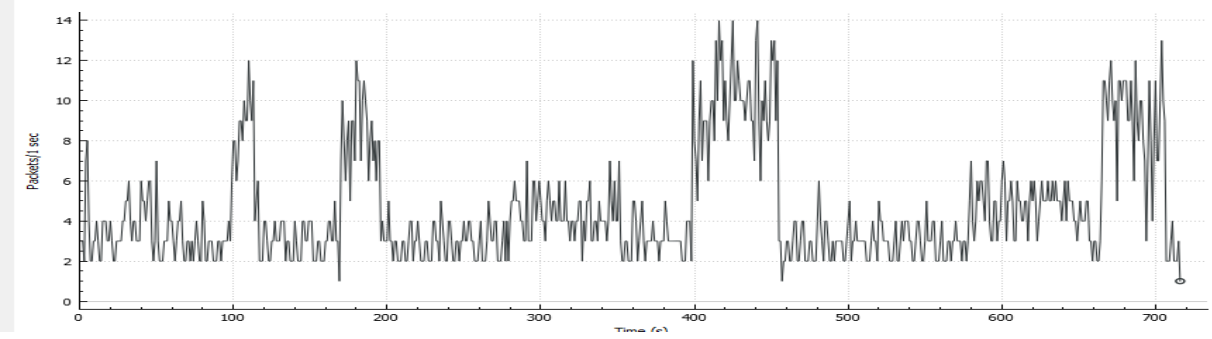


Figure 4-22 Schneider Electric (Modbus) Packets Flow

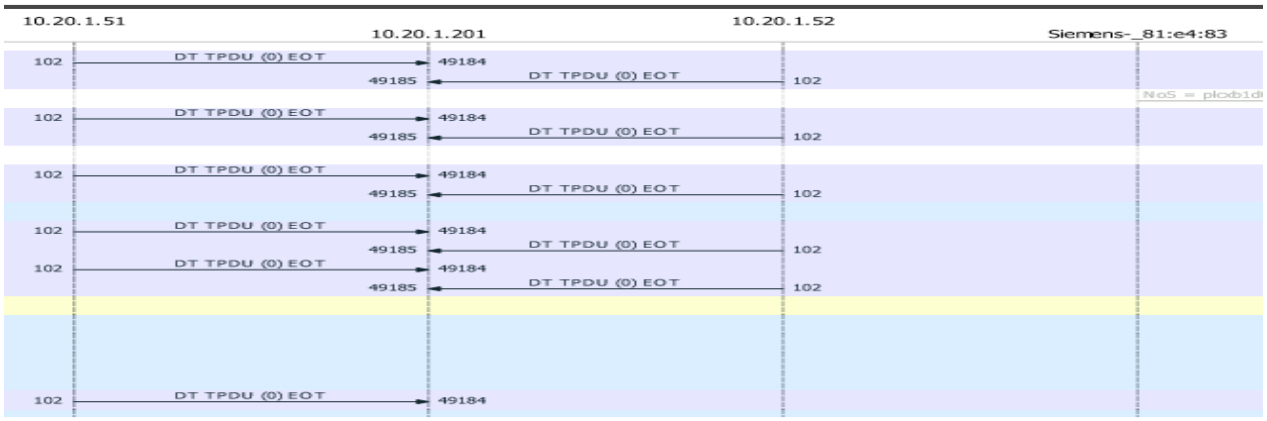


Figure 4-23 Siemens PLCs and HMI Dataflow

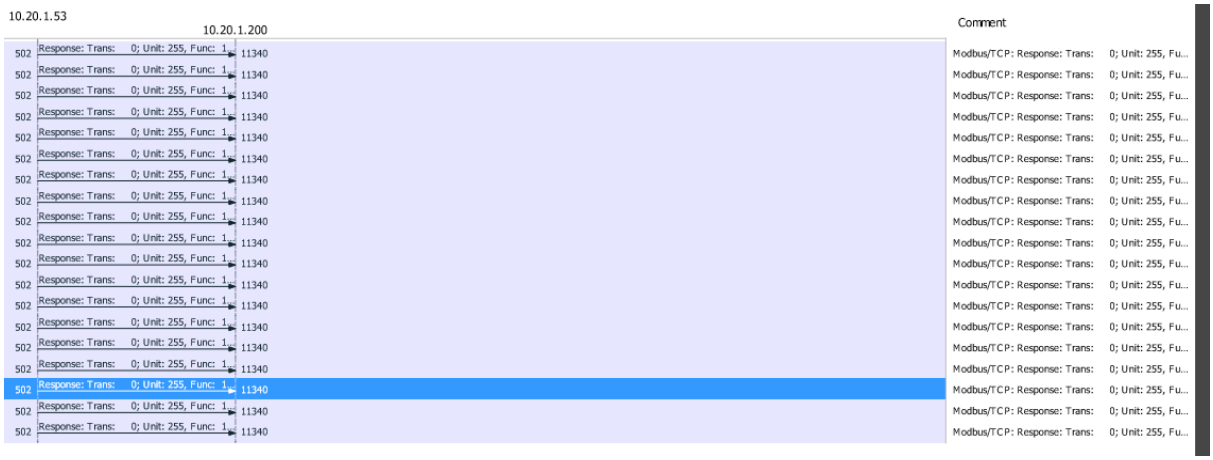


Figure 4-24 Schneider Electric PLCs and HMI Dataflow

However, Sequence is also very important in determining the flow of packets in certain direction. This will help in determining the structure of the data and handshake. As seen in figure 4.23 for the sequence dataflow from Siemens PLC and HMI. Figure 4.24, sequence dataflow from Schneider Electric PLC and HMI. Although both data are in one direction, mostly from PLC to HMI which is a response from the request made from the HMI to the PLCs. The HMI is a client and the PLC is a server and the client always initiate a request to the server, the server responds. Whether the request or the response, the information that is needed from this is to determine the sequence of the data flow for the consistency of the dataflow as seen in the sequence numbers graph below in figure 4.25 and 4.26.

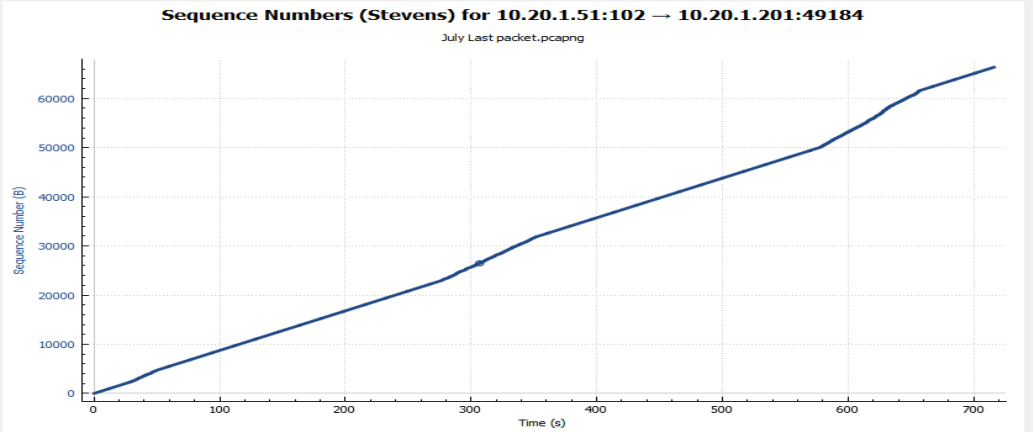


Figure 4-25 Siemens PLC 1 and HMI Packets Sequence Numbers Graph

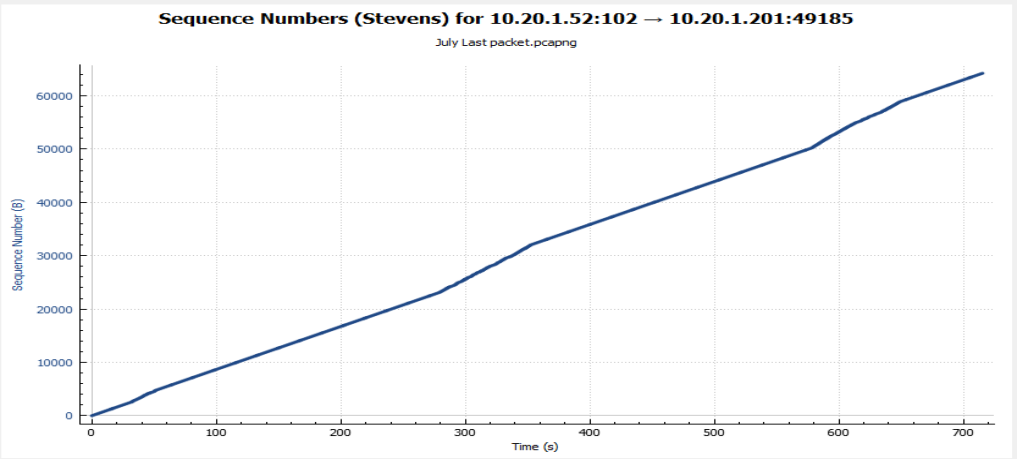


Figure 4-26 Siemens PLC 2 and HMI Packets Sequence Numbers Graph

There is a consistency in the flow of information with the Siemens device as seen in the sequence graph in figure 4.25 and 4.26. The two graphs are from the Siemens devices based on the data collected during the data collection. However, the Schneider Electric device showed a different graph view as seen in figure 4.27 below. The graph has some few delays which shows that there was no information or packet transmitted at that time. Based on the information and the analysis of the data collected, the reason was due to transmission from Siemens devices. The Siemens devices were more in number as well as more active during this time.

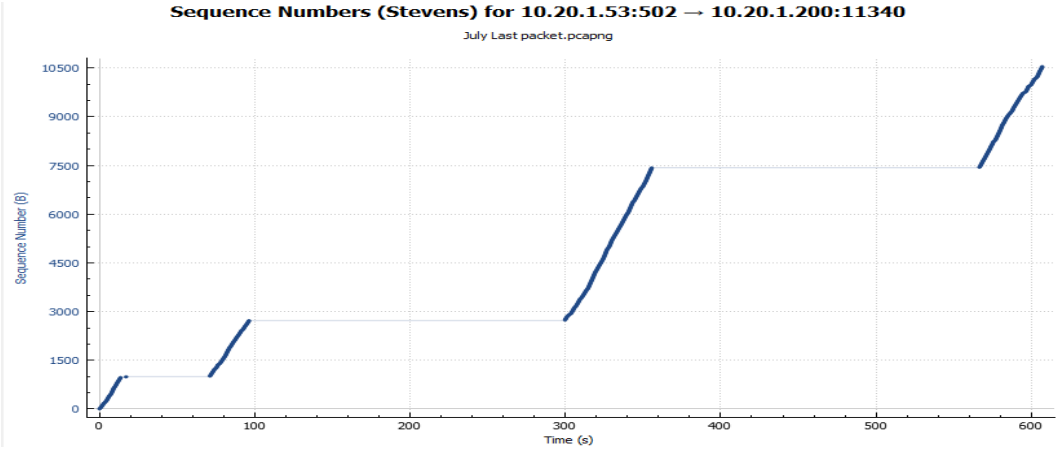


Figure 4-27 Schneider Electric PLC and HMI Packets Sequence Numbers Graph

The first part of the data was collected using Wireshark tool and a Laptop computer with Windows 10 Operating System. However, variable times durations were set for the collection of data. Data was collected for 1753.001 seconds, 1183.347 seconds and 716.321 seconds as seen in tables 4.20 to 4.22

Measurement	Captured	Displayed
Packets	3287	3287 (100.0%)
Time span, s	716.321	716.321
Average pps	4.6	4.6
Average packet size, B	117.5	117.5
Bytes	386025	386025 (100.0%)
Average bytes/s	538	538
Average bits/s	4311	4311

Table 4-20 Sample of Collected Data Properties

Measurement	Captured	Displayed
Packets	3287	880 (26.8%)
Time span, s	716.321	607.050
Average pps	4.6	1.4
Average packet size, B	117.5	66.5
Bytes	386025	58080 (15.0%)
Average bytes/s	538	95
Average bits/s	4311	765

Table 4-21 Sample of Schneider Electric Data Properties

Measurement	Captured	Displayed
Packets	3287	1919 (58.4%)
Time span, s	716.321	716.321
Average pps	4.6	2.7
Average packet size, B	117.5	122.5
Bytes	386025	235866 (61.1%)
Average bytes/s	538	329
Average bits/s	4311	2634

Table 4-22 Sample of Siemens Data Properties

The information in tables 4.20 to 4.22 shows some of the data collected and their properties. There is a consistency in the data from both Siemens (ProfiNet) and Schneider Electric (Modbus). The average packet per second for the data together was 4.6 as seen in table 4.20. This was consistent for all the data collected at different times. However, there was a bit of inconsistency with the average Modbus packet. In table 4.21, the average packet per second for the Modbus packet was 4.6 while it showed in the previous collections lesser. The reason for this was the inactivity of the system during the collection time. The time the above data that showed high number of packets per second was collected, sensors that were connected to the Schneider Electric PLC were continuously pressed at interval. Previously, that was not the case and the result was seen in the amount of data per second recoded. The Siemens PLC showed consistency in their operation as seen in table 4.22. The average packet per second was 4.6. This was the highest as the sensors were continuously pressed at interval. Others recorded, displayed 2.3 packet per second for Siemens and 1.4 for Modbus packet. The average size of the Modbus packet was 66.5 which is consistent throughout the collection period. The average packet size for the ProfiNet packet was between 122.5 and 124.5. This is still within the maximum packet that can be transmitted.

The information above indicated that the average of 4.6 packets were transmitted in second. Hence for the detection of any anomaly in the system, the latency has to be considered. For a latency of below 100 milliseconds is quite very good. This constitute the time it will take the packet to arrive before passing it to the detection engine. This might not be visible to the eyes as the time is very short. The same approach was tested for sending the data to the HBase in the cloud as seen in figure 4.28

Figure 4.28 is the information collected from the HBase for the data from the SCADA system. The information showed a total number of 9 column family that was designed for the data from the SCADA system. HBase is a NoSQL database that uses column family for data storage and retrieval. The column families here are the IP dst for destination and IP src for source. The type has a value as Scada and Scada column value is the heartbeat. The TCP both destination and source with mac address for both destination and source were recorded. These are the information that were recorded in the HBase database that can be used. However, the main purpose for collecting this data was to ascertain the time of data arrival in the system. This will help in determining the latency and the viability of using cloud system for data storage and for the algorithm.

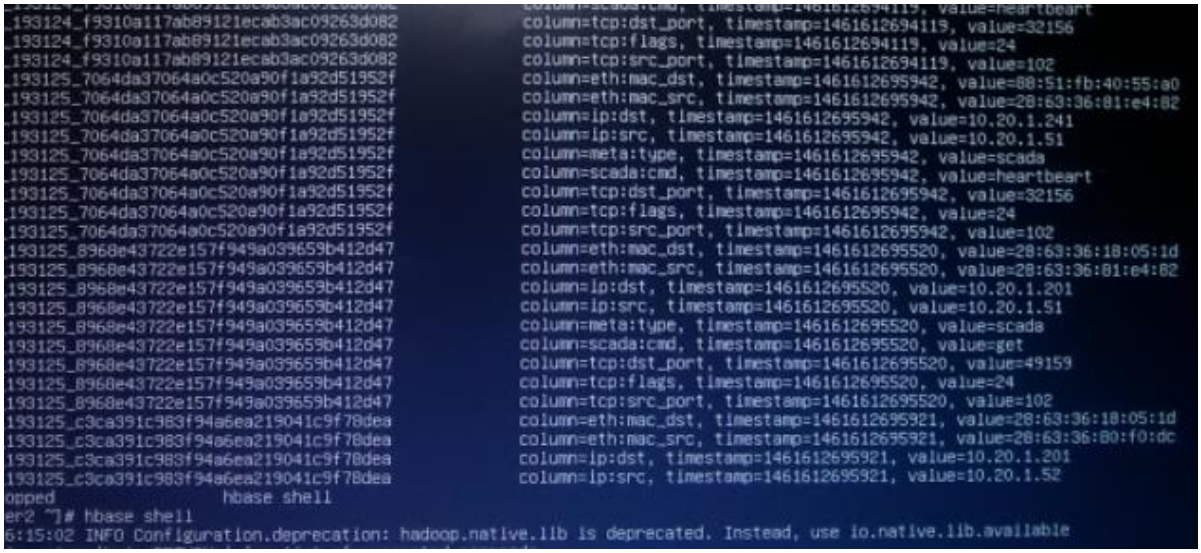


Figure 4-28 Data Stored in the HBase in the Cloud

Analysing this data that was collected over several days. The data showed some consistency with the ones collected with the Wireshark. The time was the same for the ProfiNet and Modbus data which show a latency of about 20 to 24 milliseconds. This showed that it is possible to collect and analyse data in the cloud. It was evidence that, a detection engine can be placed either in the cloud or on another system and it would not make much difference since the latency might not be visible to human eyes.

4.6 General System Model for Anomaly Detection

The tools that will be used for both in designing and modelling the system are well thought off and have been researched on. Unified Modelling Language (UML) was selected as a platform and language for designing the system in order to pinpoint any prospective issues one might encounter. UML diagram is widely recognised and has been used and reviewed by many academic scholars and researchers all around the world as a viable design and modelling approach (Khan, 2015; Bashir, Lee, Khan, Chang, and Farid, (2016)). Therefore, it is suitable for the purpose and it will yield a robust model for this project. Ethical issues have not been considered since the testing and evaluation of the product will be conducted in the lab. The lab has a setup system as well as personal computer that will be used in programming and building the model. However, the testing does not involve much contact with human being and proper training on how to use the system in the lab has been given. Therefore, the ethical issues have been taken care of and the main thing now is to concentrate on the design and modelling of the system using UML diagrams.

4.6.1 Use Case Diagram of the System

The use case diagram in figure 4.29 is an outline of the system and how the software will fulfil the requirements. In figure 4.29, the requirements that are meant to be fulfilled are the protections of the systems and recognising the forces that might cause problems in the system. The use case shows how this might be possible by identifying the possible users and how they might interact with the system in the environment as seen in figure 4.29. The use case diagram will be named Artificial Life Framework (ALF) because it defines the framework for the detection of anomalies on ICS using the birds of prey approach.

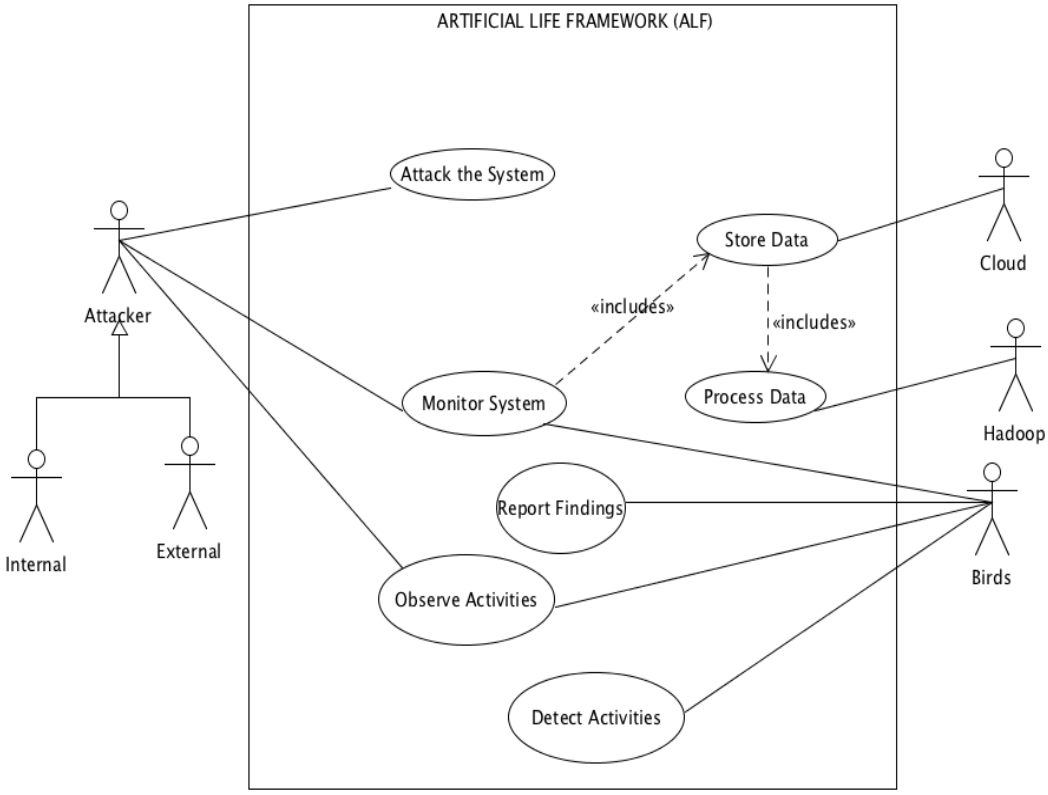


Figure 4-29 System Use Case

Figure 4.29 showed the initial use case that will be further analysed later in this project. This is actually for capturing and virtualising the system. It will give a virtual view of the system based on the use cases.

4.7 Use Case Specifications

Name of the Use Case	Artificial Life Framework
Summary Description	Using artificial (birds) for the detection of anomalies on Industrial Control System such as Supervisory Control And Data Acquisition system (SCADA)

Precondition	• Detect anything that will endanger reliability and functionality • Detection is allowed when the number of birds are six
Post Conditions or Guarantee	• Flag everything, whether good or bad • If bad is detected, increase the time to live of the group. • If good is detected, do nothing
Actors	• Attackers (internal and external) • Hadoop Framework for data storage • Cloud system for housing Hadoop and data storage • Flocks of birds for detection of anomalies
Trigger	Operator insider or an attacker initiated an operation to stop the system from running or other anomalous events.
Main Flow	Using our Train or locomotive system as an example. A gate is open that could result in the collision of the train coming from the other side. 1. Birds flying in their environment 2. Open the gate from the HMI by pressing the button on the HMI. 3. The information will be received by the birds 4. Every information is considered as predator until confirmed otherwise 5. Check the other gates and sensors information stored in the HDFS/HBase 6. Verify the reliability and functionality of the system based on the initiated action 7. The action is validated 8. The action is reported 9. Check the time-to-live 10. Remove the bird If the Time-To-Live finishes 11. Update time-to-live to default value (seconds) when anomaly is detected. 12. Every individual bird in that group (six) update their Time-To-Live 13. Update the entire flocks Time-To-Live if two groups detect anomalies 14. Update information stored in the Hadoop HDFS
Extensions	3a. Valid group of birds 3a1. The group must be six 3a2. Each bird has a specific task

	3a3. Each birds' task is based on its position 3a4. Each bird can do all the actions but only one at a time is allowed 5a. Verification of gates 5a1. Connect to the HDFS/HBase for the last information saved on the gates. 5a2. The last information showed that the opposite gate is close 5a3. Report and move on 5a4. It shows that the gate is open and there is train coming that might lead to collision. 5a5. Report and go to step 1 11a. Anomalies not detected 11a1. Time-To-Live finished and all the birds removed 11a2. Some birds have infinite Time-To-Live. They cannot be removed 11a3. Detected anomalies by two groups, default number of birds are added to the system again.

Table 4-23 Artificial Life Framework

Name of the Use Case	**Attack the System**
Summary Description	Penetrating the system from the outside or from the inside as an attacker
Precondition	• Takeover the PLC to give the wrong command to the sensors or to shutdown the PLC. • Press the wrong button on the HMI
Post Conditions or Guarantee	• On success, the information going to and from the PLC will be controlled by the attacker • Pressing the wrong button will cause functionality problems or the system to be unreliable. • Check system reliability and functionality by every action
Actors	• Internal Attacker • External Attacker
Trigger	Connectivity of SCADA system to internet as well as malicious internal behaviours.
Main Flow	1. Obtain the IP address or have physical access to the PLC 2. Connect to the PLC

	3. Firmware manipulation 4. Takeover the PLC 5. Change every request to what suits you such as when ON is pressed, change it to OFF 6. Or internal attacker pressing the wrong button that can cause the system to malfunction
Extensions	1a. Invalid IP 1a1. No access to the PLC

Table 4-24 Attack the System

Name of the Use Case	**Monitor and Observe the System**
Summary Description	Attacker monitoring the system for a possible hole in order to gain entry. The artificial life (birds) will be monitoring the system for a possible anomaly entering the system.
Precondition	• There must be activity • The system is working
Post Conditions or Guarantee	• Detect activities that will endanger the system or causes unreliability in the system
Actors	• Attacker for the reconnaissance • Birds monitoring the system for an attack
Trigger	Flow of activities in the system
Main Flow	1. Flocks of birds flying in the environment or an attacker doing the initial reconnaissance of the system 2. Monitor all the packets or protocol from the PLC. Every packet is deemed as an anomaly until proven otherwise 3. The information is stored in the HDFS/HBase and to the cloud 4. This information is used for monitoring the system
Extensions	

Table 4-25 Monitor and Observe the System

Name of the Use Case	Store and Process Data
Summary Description	The data of all the activities of the ICS is stored
Precondition	• The system is connected to the cloud for storage • The cloud infrastructure is used for configuring Hadoop • Apache spark and HBase for real time data and database are configured
Post Conditions or Guarantee	• Data flows in to the cloud • Data is stored in HDFS/HBase
Actors	• Cloud system • Hadoop
Trigger	The system generated data through activities in the system
Main Flow	1. Activities generated data 2. Data is forwarded to the PLC 3. The data is further forwarded. The Lambda architecture will duplicate the data 4. The data is streamed using apache spark for real time analysis and the other part is stored in HDFS/HBase
Extensions	1a. No activities, No data 2a. Information from Field devices such as sensors and or HMI

Table 4-26 Store and Process Data

Name of the Use Case	Report Findings
Summary Description	Report all the findings from the analysis of the activities
Precondition	• There must be activities in the system • The system must be running in order to generate activities
Post Conditions or Guarantee	• Every activity must be reported ✓ Good activity ✓ Bad activity
Actors	• Birds
Trigger	When there are activities going on in the system
Main Flow	1. Activities or events between field devices and the environment.

	2. The activities are captured and forwarded further 3. The event is analysed by the birds 4. The event is reported
Extensions	

Table 4-27 Report all the Findings

Name of the Use Case	**Detect Activities**
Summary Description	Detection of the activities or event in the system
Precondition	• The system is functional and well connected • There is activity in the system
Post Conditions or Guarantee	• Good event • Bad event or anomaly
Actors	• Birds or flock of birds
Trigger	There is an event or activities going on in the system
Main Flow	1. An event is triggered by action on the environment such as open, close etc. 2. Birds are notified, based on the valid groups starting from left to right. 3. Birds have groups 4. For a group to participate, they must be six birds in number 5. Each bird in the group has a specific task 6. Each bird in the group can do all the task but can only do one on every event 7. The tasks are located based on the bird location in the group (cell number from left to right) 8. Smallest number first and the largest last in the group 9. The first bird will check the source IP 10. Second, the destination IP 11. Third, the data length 12. Fourth, the function code 13. Fifth, verify the function code 14. Sixth, verify the reliability and functionality of the system against the event. 15. Good event, report it 16. Bad event, report it

Extensions	**3a.** Birds without group 3a1. Not considered 3a2. Cannot be involved in anything 7a. Location of birds in the system 7a1. The environments operated cell system. 7a2. Each cell can accommodate a maximum of six birds 7a3. Each bird in the cell has a location number.

Table 4-28 Detect Anomaly Activities

In order to further explain the functionality of the use cases above, class diagram will be used. The diagram will help in showing the classes that made up of a particular use case or the implementation of a use case. Several use cases have been identified as seen in figure 4.29. Use cases mainly identify the requirements for the system and the system users or the actors. The class diagram on the other hand will be used to get the user interaction and the display of the user interface. It will help in capturing the pictures of the required classes for this system.

4.8 Class Diagram

The class diagram will be used to model the entire structure of the system that will show the classes, interfaces as well as relationships between classes in the same domain and outside the domain. The main purpose here is to use these classes and relationships in resolving the problems or even locating the problem. In the case of this class diagram, the solution to the problems will be located and more prospective problems will be located, and a solution might be devised. This will be implemented using a programming language such as python.

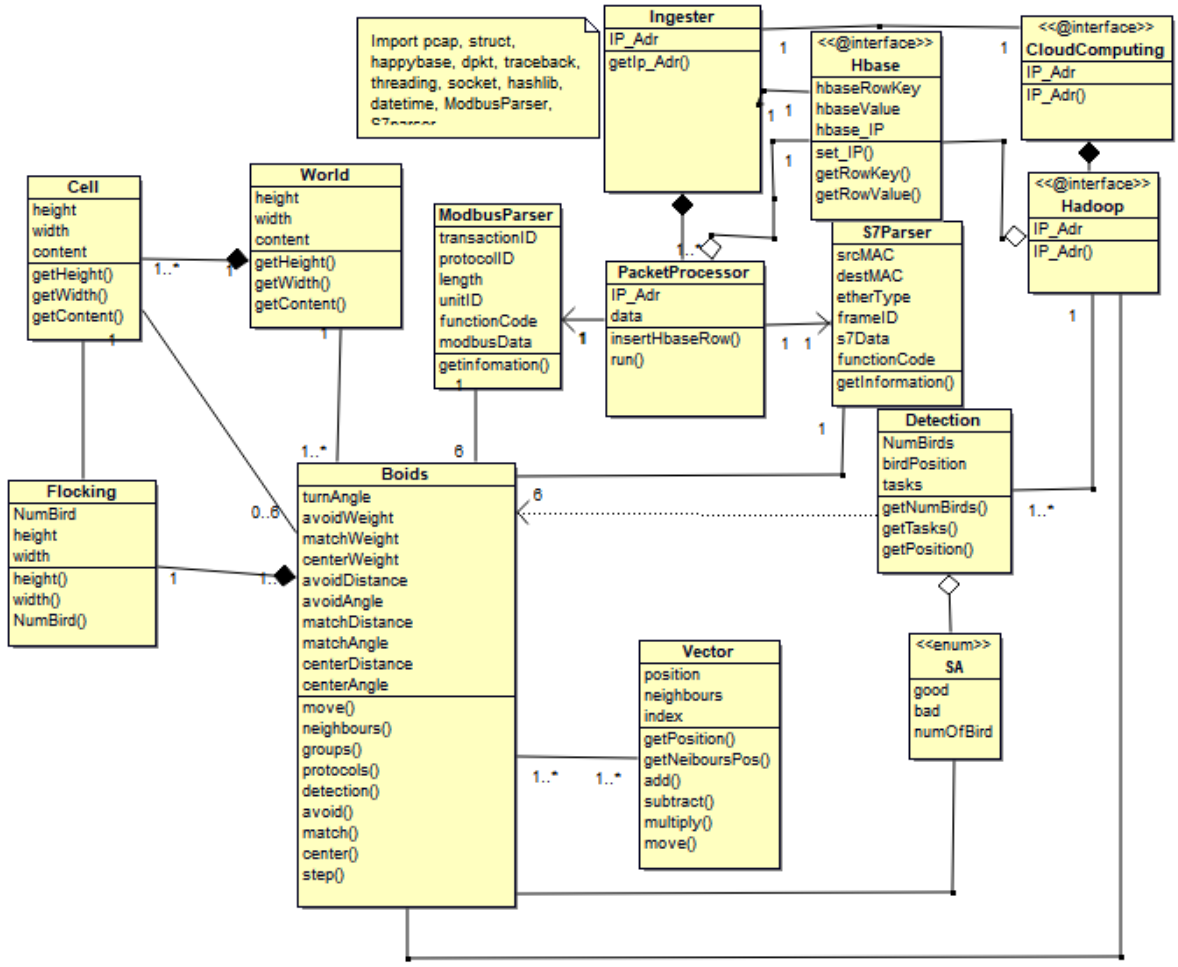


Figure 4-30 Class Diagram of the System

4.8.1 The Ingester and Packet Processor Classes

The Ingester class is the initial connection point for all other processes in the system. As an event is triggered in the system by either pressing the button on the Human Machine Interface (HMI) or through network flow, the Ingester will try a connection to the HBase, cloud and passed it on to the packet processor class. However, the main purpose of the Ingester is to ingest the package into the HBase for storage, that is the reason for trying to establish a connection.

From the class diagram above, it can be seen that HBase has an interface to the Ingester class. This will be possible by importing a module called HappyBase, which has a standard API for HBase connection. HBase uses the Thrift library from python for the gateway connection to the HBase. This facilitates both the connections as well as ingestion of data into the HBase rows and colons and even creating them when there is none as seen in figure 4.22. This will open the socket connection to the HBase Thrift server by default.

The packet processor class has a composite relationship with the Ingester class. The packet processor class is dependent on the Ingester class. The composition shows that without the Ingester class, there will be no packet processor. The Ingester class receives the initial information and establishes the connection before passing over to the packet processor for processing the information. However, the relationship or the multiplicity between them should be one to one, one packet Ingested, and one packet processed. The pseudo code for the Ingester and packet processor classes can be seen in appendix from page 9. This has a connection to HBase as a standalone (happybase, 2012);

4.8.1.1 Apache Flume

This is for the choice of following the implementation using apache flume. Apache flume is for data aggregation and transportation to HDFS or through to HBase (Apache, 2017). The flume engine aggregates data from different sources and store it in the HDFS or HBase depending on the configuration. Data flows from the sensors into the channels and sink with the HDFS or HBase sink and stored in HDFS or HBase. In this case, flume is used here to aggregate data from the environment to HBase or HDFS for further processing or storage. It is also worth noting that apache flume is an Ingester. Because this is basically to get data into HBase, the configuration of the system will continue. The configuration here will be based on ingesting data into HBase. This will start with configuring the HBase sink with the flume as seen below after downloading and extracting the flume. There are three main stages of flume process which are; source, channel and sink.

```
#Use the AsyncHBaseSink
host1.sinks.sink1.type = org.apache.flume.sink.hbase.AsyncHBaseSink
host1.sinks.sink1.type = org.apache.spark.streaming.flume.sink.SparkSink (for using spark streaming)
host1.sinks.sink1.channel = ch1
host1.sinks.sink1.table = “Table name here”
host1.sinks.sink1.columnFamily = “the column family in the table here”
host1.sinks.sink1.column = “The column in the column family to write to here”
host1.sinks.sink1.batchSize = 5000 (the maximum number of line to read and send to channel at a time)

#Use the SimpleAsyncHbaseEventSerializer that comes with Flume. This converts flume event to HBase format
host1.sinks.sink1.serializer = org.apache.flume.sink.hbase.SimpleAsyncHbaseEventSerializer
host1.sinks.sink1.serializer.incrementColumn = icol
host1.channels.ch1.type=memory
```

The configuration here will be based on table as seen in figure 4.21. It actually depends on the information that one needs to monitor or store in the HBase. The rerializer converts the events from flume into HBase format and put them in the table. The sources are configured to stream data from the PLCs and HMIs with their IP addresses as the sources and HBase as the

destination. This uses Thrift source to listen on Thrift ports with the binding hostname (IP address). This agent must be configured as follows from (Apache Flume, (2017);

```
a1.sources = r1
a1.channels = c1
a1.sources.r1.type = thrift
a1.sources.r1.channels = c1
a1.sources.r1.bind = (source IP address)
a1.sources.r1.port = (source ports)
```

from the configuration file, apache flume can be configured to channel the streaming to apache Spark. Spark is for streaming while Flume is for data aggregation and both can be linked together to stream data directly from the sources using the Thrift source. Since Spark streaming is available only through Maven Central. The configuration goes as seen below;

```
groupId = org.apache.spark
artifactId = spark-streaming-flume_2.11
version = (add version)
```

To create the input DStream, in the application code FlumeUtils must be imported. Below is the process for creating the DStream;

```
Import org.apache.spark.streaming.flume._
Val flumeStream = FlumeUtils.createPollingStream(StreamingContext, [IP address of the sink machine], [sink port]
```

4.8.2 Cloud Computing Interface

Cloud computing in the diagram is an interface to the Ingester class and it has a composite relationship with Hadoop framework. The Ingester class makes the connection to the cloud through the cloud Application Interface (API). If the connection is established, the Hadoop framework can be accessed as well as the services such as HBase, MapReduce and HDFS. Based on the diagram, HBase can work as a standalone as well as being integrated in Hadoop framework. From the theoretical standpoint for now, Cloudera will be an advantage since it has all the services already integrated for easy access. Although some of the required services were not listed in the class diagram above. However, they will be implemented and integrated in the sequence diagram. Some of the services that are required are apache Flume and apache spark. The Flume is for data ingestion into HDFS as an alternative to own Ingester class while the Spark will help for real time data analyses. Cloudera made it easy to access and integrate all these services in an environment. The pseudo code for the cloud interface (Cloudera, 2016) based on the class diagram above can be seen in the appendix page 11.

Since the approach here is a Near Real Time (NRT) data analyses, the suggested approach from Cloudera by Malaska (2015) for apache Flume will be implemented as seen above. However, with apache Spark, the detection code for the detection of anomalies in the data. This can have a low latency under 100 milliseconds. The detection code will always interact with the flume interceptor or the memory for action on whether an event is an anomaly. This system is based on Lambda architecture, therefore some actions apart from the live stream is necessary such as making decision. The flume can interact with the batch storage or HBase for decision also. Malaska (2015) reported that the HBase could give out information within 4-25 millisecond based on the type of network. Below in figure 4.31 is the dataflow architecture with the lambda as described in Okeke and Blyth (2016) as No-Trust-Zone architecture (NTZ). Data supposed to flow from the environment into the HDFS and the streaming data will be through the Spark. The algorithm will be checking the incoming information with the heartbeat of the one stored in the system. The heartbeat here is in millisecond for checking between the stored data and the event in the stream. The algorithm will be integrated in the Spark as it offers machine learning implementation technique. See the appendix for more detailed explanation on this, appendix.

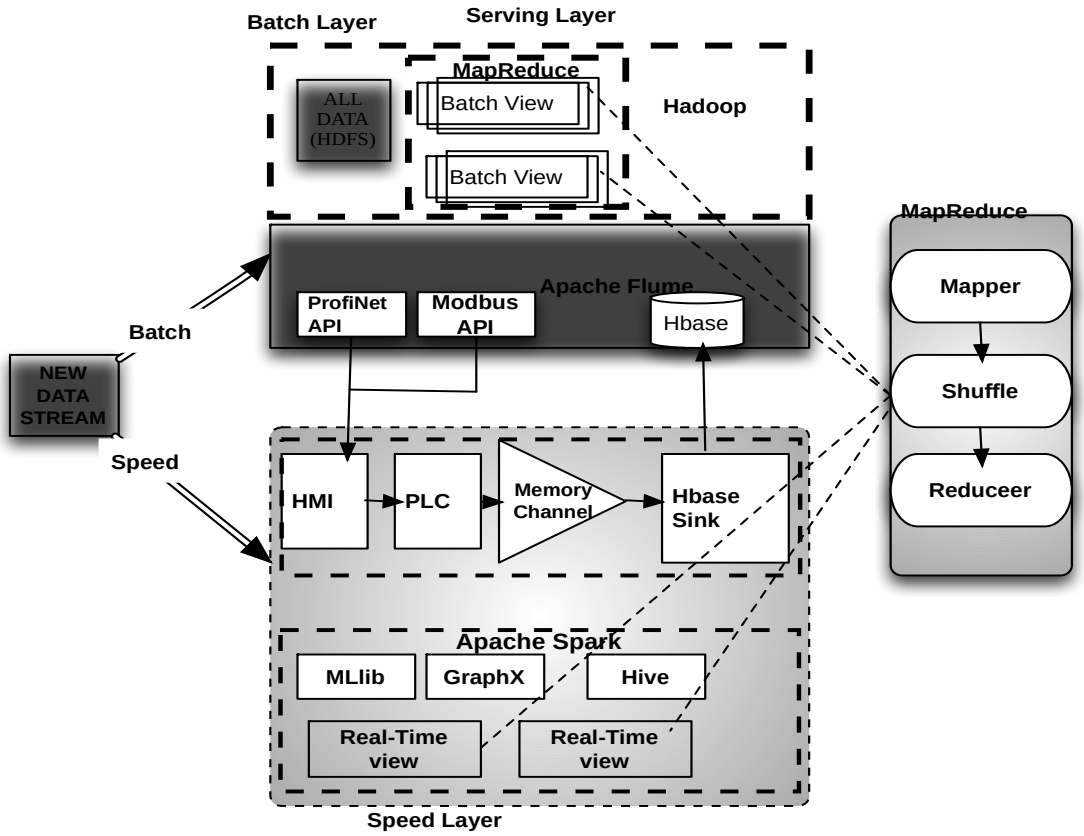


Figure 4-31 Dataflow and System Architecture

4.8.3 S7Parser Class

The purpose of this class is to find a way of parsing the information in the protocol or structuring the information for easy readability by other programs. This will help in knowing the information that is required by the detection class and others in other to detect anomalies in the system. The parser has a relationship with the packet processor class. The packet processor gets the required information from the parser in order to identify the packet and their structures. The S7 parser pseudo code for this project can be found in the appendix from page 11.

4.8.4 ModbusPaser Class

Modbus is an open protocol, meaning that it can be used and customized by other vendors and is available to anyone. This has made it popular among manufacturers and developers. This is one of the reason Modbus devices was chosen as it is one of the mostly used product in industrial automation which in 2011 contributed to 22% of the market share based on the report from Control Engineering (2011). Another and most important reason of chosen this protocol was that at the moment, it is the protocol that was available in the device during this study. The protocol is a master slave protocol that has a function code which carries the instruction that will be implemented or acted upon. The structure of the parser that will show the information that will be passed over to the birds for detection of anomalies can be seen in appendix from page 14. It represents the pseudocode for the Modbus protocol that will passed on.

From the Modbus point of view, protocol ID plays a very important role as it will help in identifying the protocol, whether it is S7 protocol or Modbus protocol. This is what will be inserted in the Packet Processor class in order to identify the protocol.

4.8.5 World Class /Environment

The world class is the environment where the birds fly around and have their being as an artificial life. This is their world and it must have dimensions such as height and width. These parameters will help in defining the environment and the expected behaviors in the environment. Knowing the dimensions will help in determining the position of the birds in the environment as well as the neighbors. The number of neighbors is limited to six and their positions will be very vital in detection processes. Their position at the time of analyses will

determine their function as to whether they will be responsible for the transactionID, protocolID and others. The pseudocode of the world class for a two-dimensional world can be seen in appendix from page 15.

4.8.6 Cell Class

The cell class has a composite relationship with the world class as seen in the class diagram above. This shows that, it goes with the world class and cannot exist on its own. However, the cell must be the exact measurement of the world based on the world width and height. A cell can only contain a maximum of six birds and the bird's positions in the cell is very important for their allocation of functions at that time. The pseudocode of the cell class can be seen in appendix from page 16.

4.8.7 Boids Class

The Boids class or the bird class is taken from the description given by Reynolds as described in Reynolds (2015). The Boids class is a simulation of the birds in an environment which will show their position as well as their velocity and neighbors. The world class has been created with birds in it, therefore this is the real construction of the birds and some of its useful characteristics for this project as stated earlier. The birds are the real agents that will interact with each other in their world in order to detect a predator. Because their world contains cells, there is no need of monitoring their neighbors. However, they will move to another cells which is none determinant and random. There are three important rules according to Reynolds (2015) which are; orientation, Cohesion and attraction. These are the rules for maintaining flocking as a group. Please refer to appendix page 16 for the pseudocode of this class.

4.8.8 Pair Class

The pair class can be used to find the pair of birds in a cell as well as adding them and subtracting them in a cell. This might be used as a vital tool for adding and subtracting pairs in a cell. The method can be defined in order to help in counting the groups available. It is possible to do without this class and add the function to the detection class or the world class where the neighbors' method is defined. Please refer to appendix from page 18 for the pseudocode of the pair class.

4.8.9 Vector class

The vector class will help in determining the direction and distance as the vector has to do with the magnitude and direction. If a straight line is drawn from point A to point B in order to reach the C. The C is the magnitude while the angle it will form, if a right angle is drawn, will be the direction of the vector, which is the direction of movement. This will help in determining the direction and speed or velocity of the birds in order to maintain the three rules stated in the Boids class. The vector class pseudocode can be found in appendix page 18.

4.8.10 Flocking class

Flocking class might not necessarily be needed but for the purpose of demonstration and for the experiments, it will be added. The flocking will be the main class that will call other classes and modules. Here the number of birds will be set as well as the width and height of their world. Using a while loop to iterate through the process of their movements from one location to the other. With regards to iteration, this will be continuous. Please refer to appendix page 19 for the pseudocode of flocking class.

4.8.11 Detection Class

The detection class has an aggregate interface with the enumeration class “SA”. The SA stands for situational awareness, which shows the situation on the inside. The SA will show whether something is bad or good and the number of birds that are still remaining in the system. When a group of birds detect an anomaly, it will reflect on the SA and their time to live will increase as well as when two groups detect anomalies, the whole flocks will update their time to live. The detection is based on the network traffic as well as the protocols. There are some parameters that are passed on to the detection class in order to check for any anomalies based on the protocol. Modbus and S7 protocols have different structures, therefore their packet structure needs to be understood by the detection class and the birds. The pseudocode representing the detection class can be seen in appendix from page 19.

4.9 Sequence Diagram of the System:

The sequence diagram will outline in the diagram the sequence of event that is expected in this project. This will show their sequence of occurrences. The diagram will help in determining and to identify more of the required resources for the project as seen in figure 4.32 and 4.33.

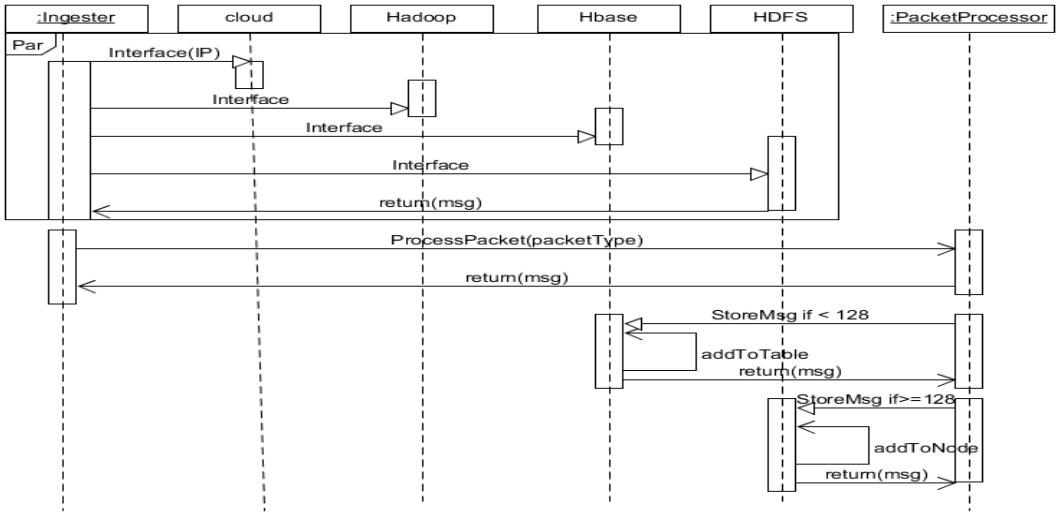


Figure 4-32 Sequence diagram of the initial connection to all the peripheral

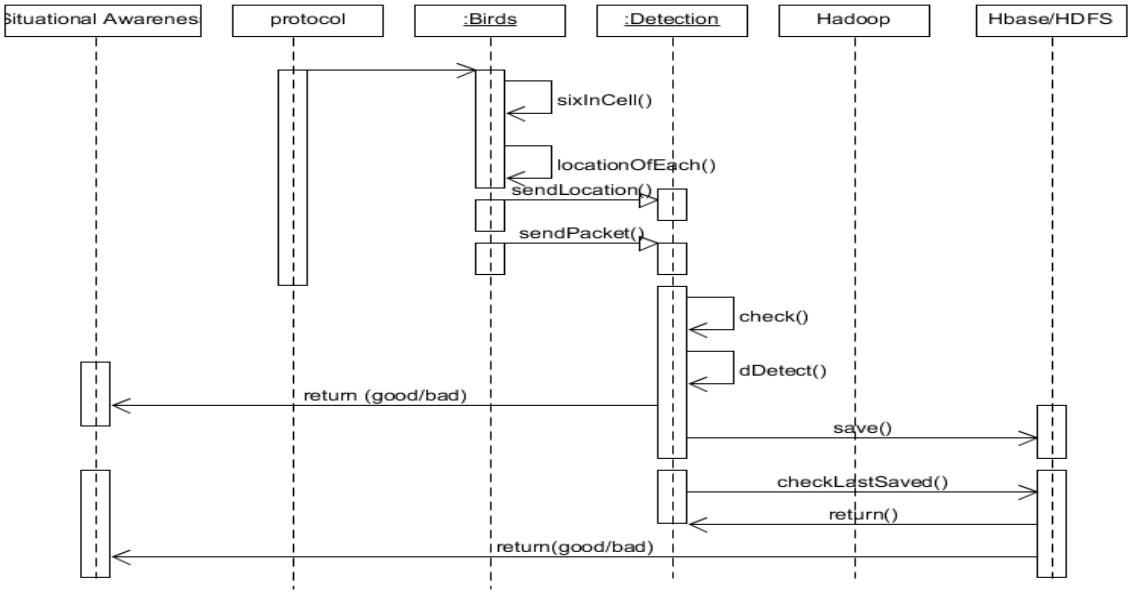


Figure 4-33 Sequence diagram for the detection approach

Summary

The design of the system was to locate some key features of the system that will assist in modelling and integrating the model in the environment. Through data collection, the type of data used in the system was determined for both Schneider Electric devices and Siemens devices. A common ground was found between the two companies; the data type for both are mainly common Boolean. Because the sensors were connected through the PLC, the command given to it was either to open or to close. It is either in the state of ON or OFF which is true or false state as seen in figures 4.16, and 4.17 as well as tables 4.13 and 4.24. Through the data collected, the average data frequency was determined as seen in tables 4.20, 4.21 and 4.22. This was a vital information as well as others in modelling the system for the detection of anomalies.

The general system communication approach in terms of the bird's prey, the cloud computing and the Hadoop framework was designed in form of model. This was modelled using the UML diagram. The initial use case diagram was to gather information that is needed for the modelling and to virtualize the requirements. This was followed by class diagram that shows the connections and the display of the object-oriented approach. This was for the simulation of the bird's prey in their environment as well as for the general modelling of the system. The environment in the class diagram was represented as the world. The world has cells that can only contain the maximum of six birds. Ingester class is to collect the data from the system and processed by the packet processor class and passed on to other classes for further analysis. This can be seen in the sequence diagrams in figures 4.32 and 4.33. The class diagram was decomposed in the sequence diagram in order to virtualize the flow of information. Evidences showed that information can flow from the system to the cloud for the general model. However, information collected during the data collection showed that the latency for the data flow was below 100 milliseconds. For a latency below 100 milliseconds, it is a good sign that cloud and Hadoop can be well utilised for such system. Chapter 5 will be further modelling of the system using petri nets formalism.

CHAPTER V

5 MODEL FOR THE LOCOMOTIVE SYSTEM

Chapter four went into the general model introduction using Unified Modelling Language (UML) which was defined by Rumbaugh, Jacobson and Booch, (2004). The UML language was used to capture all the requirements needed for the system and for modelling the system. This was evidence in both the use case diagram and definition as well as the class diagram and others. This information is needed in modelling a complex system such as the flocks of bird's collective behaviour or the SCADA system. UML was used as a step in having a broad view of the system in terms of the components communication as well as their activities. Although chapter four is not about the modelling of the system, rather it was about the design of the system which flows into the first models. The diagram was a way of virtualising the system and its functionalities. This chapter is about the further model this project will also produce as an artifact. Hence it is another product of this project and this model will be a way of communicating the idea, virtualising, testing and evaluating it. Therefore, the idea will be modelled using Petri nets formalism.

There are various types of Petri nets formalism such as coloured Petri nets, Jensen (1987), Reference Nets from Kummer (2002), Nets Within Nets by Lomazova (2000). These are all the families of Petri nets and the difference between each of them is their semantics. Nets within nets semantics is for a proper structure of a token in a net. Hence a token is a net and can move and fire themselves. Reference nets deals with synchronous channel, and tokens can represent a net, or a token can be a reference to a net. The semantic of reference nets is similar to coloured Petri nets but the main difference is on the token as explained earlier. Coloured Petri nets on the other hand comprises of tokens, arc, transitions and places. It is used to model events that occurs in a system or events that can result in a change in a system. Events occurrence normally changes the state.

This chapter is not about the proving or demonstration of the best suited modelling language. However, the purpose is to find out the best way of representing the idea as well as best possible way of presenting the idea without losing the properties or important information. Coloured Petri net was deemed best in explaining, modelling and analysing the idea. Coloured Petri nets is a graphical modelling language that has been used to model concurrency, complex systems, distributed systems, synchronisation, data manipulation and among others (Jensen, 1997;

Kristensen, Christensen and Jensen, 1998). There are quite a number of tools that can be used such as CPN tool (CPN tool, 2015), Platform Independent Petri Nets Editor (PIPE, 2013) to simulate the model as well as performing some analysis. This project will use CPN tool for modelling and analysis of the model. CPN tool has come to a maturity since the initial development in 1989 as Design CPN tool (Christensen, Jorgensen, Kristensen, 1997). CPN tool has been used in obtaining quality results from a model (Lindstrom, and Wells, 1999). The record showed that the CPN tool can be used to obtain quality results of a model and analysis. The model will come in form of bottom up approach. That means the methodology that will be followed for the modelling.

5.1 Modelling Approach

The modelling approach that will be adopted here is Agile for the effective modelling and documentation (Ambler, 2017). Agile methodology deals on step by step approach in modelling and designing things. This approach has proven effective compared to the waterfall which is very rigid and depends on the final product at the end (McCormick, 2012). The agile modelling approach has been used successfully in modelling the cellular signalling (Danos, 2007; Thul, Thurley and Falcke, 2009). The agile modelling approach or agile in general deals with project in either top down or bottom up approach. This means that projects are done in stages, which can involve multiple people working on a project from different location as well as in the same location. Projects are broken down into smaller workable unites, work on it, analysed and at the end brought together. Projects that follow this approach has shown that the approach exposes mistakes easily compared to waterfall. Hence any mistake in the project or if the project is not viable, it will be rectified early.

The unit of analysis in this study is the train system as discovered in the research methodology, see appendix. Discovering this in the early stage is for the data collection in the initial stage. The data collection in chapter four was to determine information that is needed for the model. Hence the modelling will proceed by looking into the bird's model as the unit of analysis as well as the train system. The modelling will start with the general view of the bird prey model for all ICS and proceed further to the specific model. The specific model that will be specifically for the model of the SCADA system introduced in chapter four.

5.1.1 Visualising the Bird Model

Figure 5.1 represents a system where everything is a bird, and it shows how birds with the unique ID that begins with B are connected to others such as HMI, PLC and other sensors. Everything apart from bird B is a sensor, because information is being retrieved from them about their state as well as other information. These birds have a unique identification (ID) beginning with B1 to B18. The number can be more depending on the system. Each bird based on the information in table 5.1 can monitor on average of 3-4 devices on a system. These devices are connected to the birds and information is retrieved from them with regards to their state, status, based on the Decision algorithm that will be constructed. Although these devices are regarded as birds, their duty is to produce information about their state, status and other required information. This information will help in making an informed decision about their condition. The information will tell whether the device is dead, alive as well as the condition of the device at that time. Dead can mean switched off or malfunctioning of the system. Table 5.1 shows all the connection for each bird. Some of these birds are connected to maximum of five devices and some four, three, two and some are not connected to any device.

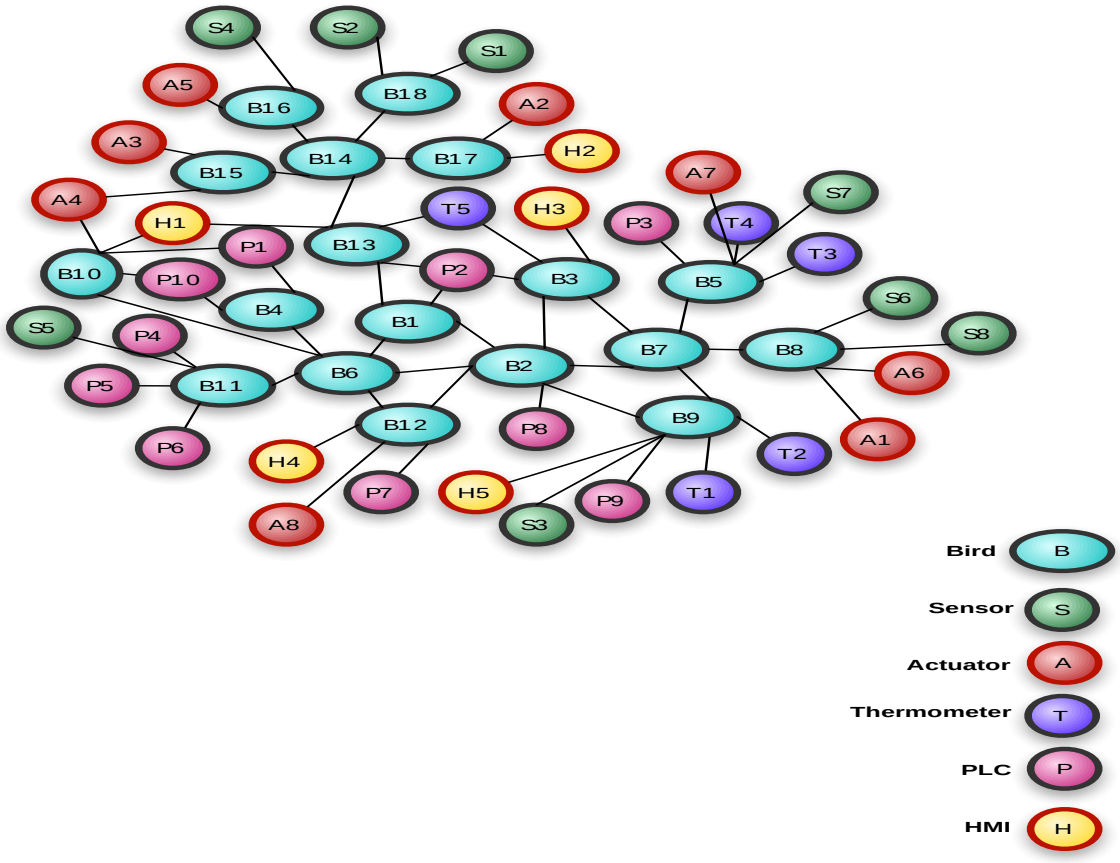


Figure 5-1 Birds Connections and Communication

Bird ID	S-ID	A-ID	T-ID	P-ID	H-ID
B1	X	X	X	P2	X
B2	X	X	X	P8	X
B3	X	X	T5	P2	H3
B4	X	X	X	P1 P10	X
B5	S7	A7	T3 T4	P3	X
B6	X	X	X	X	X
B7	X	X	X	X	X
B8	S6 S8	A1 A6	X	X	X
B9	S3	X	T1 T12	P9	H5
B10	X	A4	X	P1 P10	H1
B11	S5	X	X	P4 P5 P6	X
B12	X	A8	X	P7	H4
B13	X	X	T5	P2	H1
B14	X	X	X	X	X
B15	X	A3 A4	X	X	X
B16	S4	A5	X	X	X
B17	X	A2	X	X	H2
B18	S1 S2	X	X	X	X

Table 5-1 Lookup table of the Connections

5.1.2 Information Transfer Among Birds

One of the important aspects of this model is the information transfer among birds or the connected devices. Birds communicate through their actions and their actions shows that information travels from one bird to another. How this information is being transferred will be communicated in this model. This will show the possibility of reaching other bird from one point to the other. Information transfer is very vital because it will show why birds move in an aerodynamic as they are maintaining their rules of cohesion, attraction and repulsion among themselves. If this information can travel freely, it is an indication that they can detect and responds to any intrusion accordingly. If the bird "B1" received information and the bird "B18" receive the information immediately, it shows the effectiveness of information transfer. Figure

5.1 will be tested based on the information transfer between the birds. This will be a form of testing the reachability of the bird in another side.

1. From bird B18 to bird B9. This will be in form of a set and the process are as follows B18, B9 = {B14, B13, B1, B2}
2. B16, B12 = {B14, B13, B1, B6}
3. B15, B10 = {B14, B13, B1, B6}
4. B11, B8 = {B6, B2, B7, B8}
5. B13, B5 = {B1, B2, B7} or {B1, B2, B3, B7}
6. B5, B10 = {B7, B2, B6}

These shows that any bird in the flock can be reach from anywhere and from any bird in the same flock. These are groups that forms the flocks and more birds can be added to the system if the number of devices in the system is increasing or reduced, if the number of devices in system is reducing.

The bird design in the algorithm 1 below started with the initial 200 birds. This is based on the size of the system. The system with 10 sensors and 3 PLCs is a very small system. However, that can also represent the entire SCADA system as the needed components were available, although in a small number. The TTL in this algorithm is 20000 milliseconds which amounts to 20 seconds. Inactivity of any bird for 20 seconds will disable the process of that particular bird and save the memory and the processing power. Although this is specifically for this particular 200 birds. The more the number of birds, the lesser their TTL.

Bird (bird)
NumberOfBirds = 200
Time_To_Live (TTL) = 20000ms
∞ TTL: [B1, B2 …. B18] + [A1, A2…An] + [S1, S2…Sn] + [T1, T2…Tn] + [H1, H2…Hn] + [P1+P2…Pn]
Alive = 1
Dead = 0
ID: B1, B2, B2, B4 B5……Bn
Status: 1

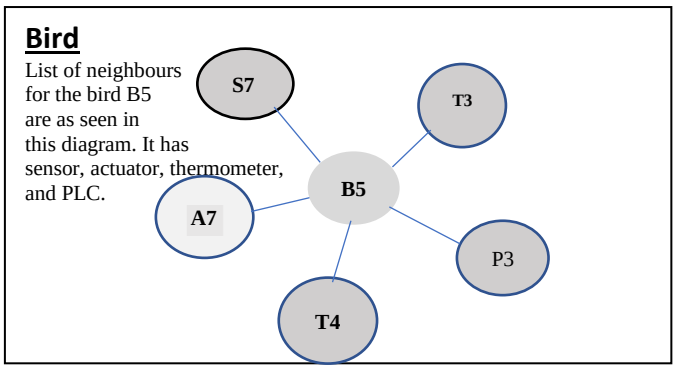


Figure 5-2 Group

State: {Scanning || Idling}
##using the bird with the unique ID B5 as an example here. From the table 5.1, the neighbours are
ListOfNeighbours: {S7, A7, T3, T4, P3}

Decision Algorithm (DA)
Begin:

```
For bird in the environment Do:
        # Checking the properties of a bird; STATE, TTL.
        check TTL
        If TTL = 0
                deactivate ()
        Else if TTL > 0 && state = scanning
                TTL = default TTL
        End if
End for

For Neighbour in the ListOfNeighbours Do:
        compare state (broadcast.birdID)
        If status = 0
                Report neighbour
        else if status = 1
                Scan Neighbour
                        if latency >= 30ms
                                Report Neighbour
                        End if
        End if
End for
End
```

Algorithm 1 General Algorithm of the Birds

Bird (HMI)

Alive = 1
Dead = 0
ID: H1, H2, H3……...Hn
Status: 1
State: {ON || OFF}
##using the HMI with the unique ID H1 as an example here. From the table 5.1, the neighbours are
ListOfNeighbours: {B10, B13}

Decision Algorithm (DA)

```
Begin:
        for bird HMI in the environment Do:
                get the state
                get the status
                broadcast (birdID, state, status)
End
```

Algorithm 2 General Algorithm of the Field Devices (bird) Model

5.1.3 Bird and Its Decision Algorithm

The information in figure 5.2 is the general view of the model where sensors and other field devices are regarded as birds and counted as neighbours. This was used as an example in trying to illustrate the view of the birds in a system. The characteristics seen in the single bird can be seen in all the birds except that some birds have infinite TTL and some have the maximum of 20000ms (limited) TTL, which amounts to 20 seconds. The information here with regards to the time is not static, it can be changed based on the environment. This is based on the small-

scale environment available for this project as seen in chapter 4. Birds without infinite TTL dies as their TTL finishes. Death here means, deactivation or muting of the process in the system. When the process is deactivated, it frees the memory space and allow the system to be faster. In the global declaration, sensors such as HMI, PLC and others have infinite TTL because they don't die. However, the birds with the ID starting with B can give birth to new birds and they are connected to field devices. There are definite number of birds that can be in the system based on the size of the system. These initial numbers can be identified with their unique ID as seen in figure 5.1 and 5.2. In figure 5.1, the number of birds with ID starting with B are 18 and these ones can spawn because they have infinite TTL. The initial number declared are 200, which more can be added into the number of birds with infinite TTL if their activities are vital. The reason for having these is to give room for expansion in the system, flocks can be added to other flocks in order to form a bigger flock.

The decision algorithm for the bird was to check the TTL, neighbours as well as the groups. The decision algorithm is responsible for getting information on the state of the neighbours and broadcasting and reporting it to the operator through situational awareness. Through broadcasting, information will be compared based on the unique ID of the device. The devices that are dead or not functioning will report the state when requested by the bird as well as the ones that are working.

5.2 Modelling a Lone Bird with Petri Nets

Modelling a lone bird in this instance is to virtualise the properties in a bird. A lone bird is a bird that is on its own, not flocking with other birds. It has all the properties seen in the flock of birds. Single bird can scan, detect predator and eat on its own. Understanding the qualities of a bird will make it easy to explain their properties as a flock. In section 5.1, birds can only do two things; scanning or idling. The literature review in chapter 3 showed that the detection in birds is through their scanning frequency (Dehn, 1990). Hence scanning is very important for the detection of predator. The more the number of scanning frequency, the more probable any predator will be detected. This approach was broken down in more understandable way by Okeke and Blyth (2016) as follow;

P_{GD} = 1 -e^{-VTN}
P = Probability
V = Scanning frequency
G = group
N = the size of the group

T = time

Probability of the group detection of predator is dependent on the scanning frequency. As earlier mentioned, a bird is either idling or scanning. Group detection is also dependent on the active scanning birds, the more the number the less individual scanning and more eating or idling and less in the group, the more scanning and less idling. This shows that the more the number of birds in the group, the less the scanning frequency and the more the probability of detecting any predator. Detection is possible using time and scanning frequency in order to detect whether a node is down. This can be done through sending and scanning for the heartbeat of the system. The heartbeat here is the scanning frequency of the system in milliseconds. A bird can either be idling or scanning, idling means that it is doing nothing while scanning is the heartbeat. This single bird approach can be modelled in petri nets as follows.

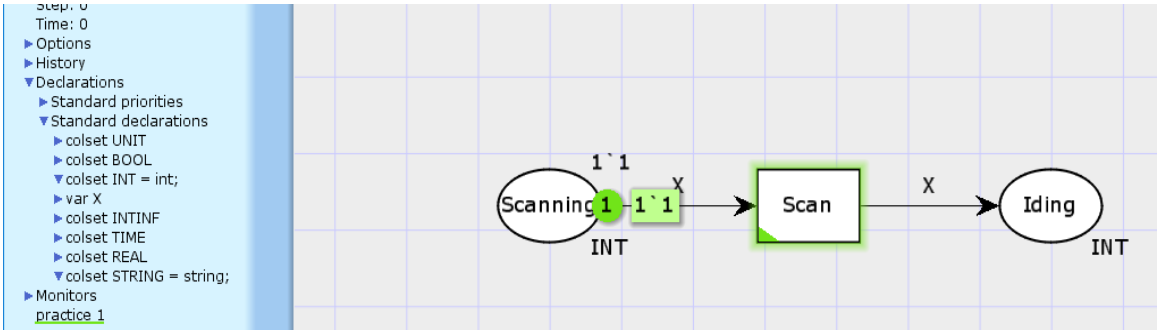


Figure 5-3 Single Bird approach

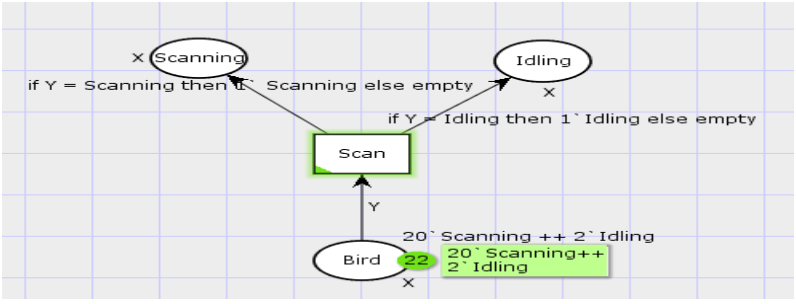


Figure 5-4 Single Bird Model of Scanning or Idling

Figure 5.3 and 5.4, showed that bird can either be scanning or idling. Bird cannot do both at the same time, because when a bird is scanning, it cannot be idling. Scanning means that the bird is watching out for any predator and the literature review showed that scanning frequency is one of the antipredator approach use by birds. The more the scanning frequency, the more the possibility of detecting any predator. In other hand, this shows that idling or eating time is not an effective time for the detection as birds are busy with the food. Idling time in this model will be regarded as the time birds are not doing anything. In other way, they are not detecting, neither are they flying. Figure 5.3 located these two most important activities in the life of a

bird. It is made up of two places, “Scanning” and “Idling” as well as one transition which is the action location. The transition is the action to scan, but this can only happen if there is a token in the place “Scanning”. One token (1`1) was placed in the place “Scanning” in figure 5.3 that activated the transition to fire. The green sign shows that the transition is ready to fire which will result in the token from the place “Scanning” being consumed and another token will be placed in the place “Idling”. The places are of the type integer that resulted in data number 1. The variable x was used for the data type which represents the integer for the data going through the arc. Hence x was used as an arc inscription for the data type.

Figure 5.3 was further developed to include birds that are either scanning or idling as shown in figure 5.4. It shows 20 scanning and two idling times for the bird and as before, bird can only scan or be idle and not both at the same time. However, the ratio of scanning to idling was 10:1 as seen in figure 5.4. The ratio here was a demonstration of how important scanning is for the detection of anomalies. The places were named bird and they contain these two types of tokens; Idling and scanning. The tokens in these places were 20 scanning and 2 idling tokens, which activated the transition to fire for 22 times. The if statement on the arc inscription allows the scanning time to go to the scanning place and the idling or eating to the idling place. As stated, if the bird is not scanning or idling, do nothing or allow it to be empty. This was a further development of the previous model that separates the two activities of the bird. Hence, scanning times and idling times do not go to the same place when fired. This is a demonstration of one of the properties of the bird model, scanning and idling. Scanning which signifies searching while idling signifies the time at which no activity is happening. This can also represent the time the bird is disabled.

5.2.1 Spawn and Die Model

Figures 5.3 and 5.4 are the introduction of the important activities of a bird which are idling and scanning. However, there are other important activities that are needed in order to integrate the bird into a system and for effectiveness as an artificial life. One of the characteristics of a life based on chapter 3 is the ability to give birth as well as die. Modelling this approach into the system will be a major improvement on figure 5.3 and 5.4. Scanning times can be increased and the idling time can be removed.

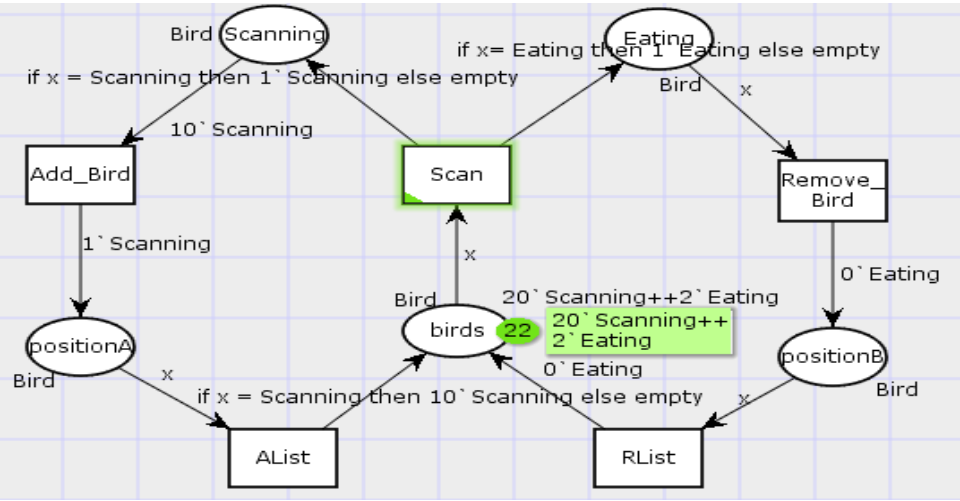


Figure 5-5 Spawn and Die Model

Figure 5.5 is a model that demonstrates the extension of figure 5.4 to include addition and removal of birds in the system. The eating or idling as earlier shown in figure 5.4, is a state where bird is doing nothing, and it is as good as dead. Hence the state is regarded as dead state, or the state where there is no information coming from the device. This means that the device is not responsive, showing that there is a problem in the system. This can be referred to devices such as HMI, PLC and other sensors in the system. However, it can also be for a bird with TTL that has finished. When a bird TTL is finished, for the ones without infinite TTL, they die and are disabled or removed from the system as demonstrated on the left side of figure 5.5. The right side represents the active birds as well as scanning frequency. The arc inscription that went out from the Scanning place shows the number 10 for the scanning. That means the frequency of the packets that are coming in. When the number are 10 packets per seconds, 10 birds should be added to the system and the circles goes on that way. The AList on the right side means the added list while the RList on the left side means the removal from the list. The number 10 for the scanning is not permanent and can be adjusted based on the need. Based on the information in chapter four, the data collected showed the maximum of 4.6 packets per second. However, this can be amended based on the system and they type of device (PLC) and communication protocol it uses.

5.2.2 Adding Groups

Figure 5.1 and table 5.1 showed what having group meant. Birds watch each other as in a group and in the same way information travels within the group to other groups. The flow of this information is as a result of connection from one bird to the rest of the flock as demonstrated in section 5.1.1. Figure 5.1 showed that HMIs, PLCs and other sensors are connected to birds

and are regarded as birds themselves. These devices can only send information with regard to their state and status to the bird in blue colour. The birds in blue colour requests or pull this information from these devices as a way of interaction and these devices broadcast this information with their unique ID. Their unique ID will make it easy to identify them in the group. The same way, some birds can belong to more than one group as well as some other devices (as birds) can be connected to more than one bird. This group is very important for information retrieval and the integration of it in the model is the next step as seen in figure 5.6.

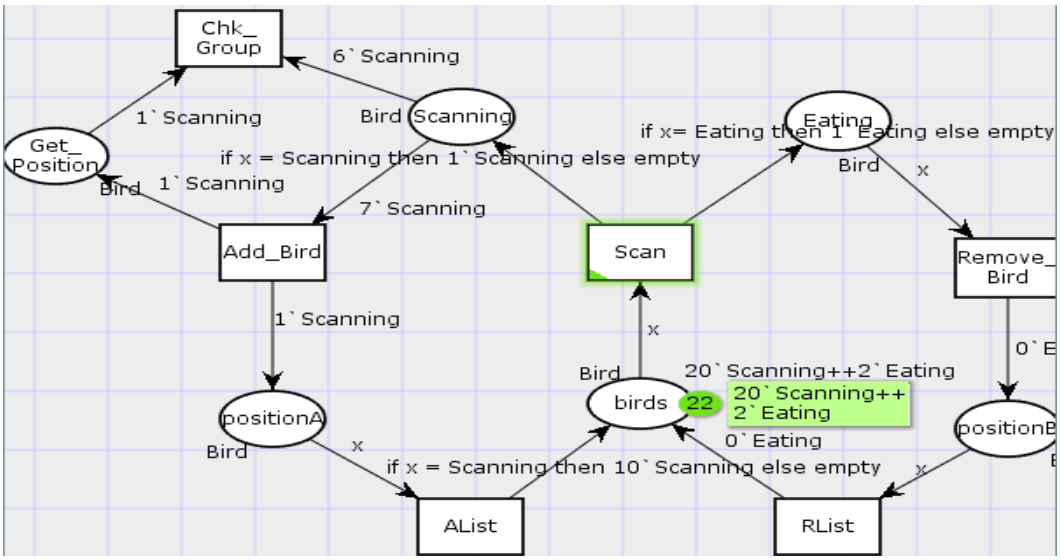


Figure 5-6 Group Added

5.2.2.1 Unfolding the Group

Group has been added to the model to reflect the presence of the group or groups within the flock. The place scanning represents the neighbours which involves all the 20 birds in the flock. However, group has a maximum of six connections, which is represented with an arc inscription from the place scanning to the transition Chk_Group (check group). The transition is activated when the six is met. This six are counted from the place scanning and once that is met, the transition will be activated with the position of each bird in the group. Their position will be their unique ID to make it easy to identify any bird in the group. When a system is down, the operator can easily refer to the lookup table for the particular device in question based on their unique ID as seen in table 5.1. The lookup table which can be digitalised for the unique ID and location virtualization. However, figure 5.7 will be the unfolding of the group.

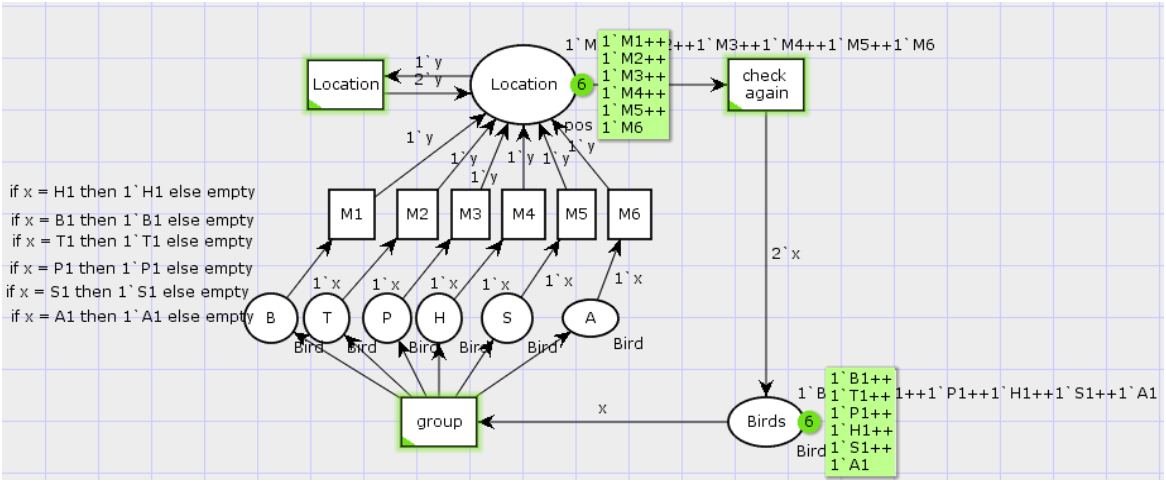


Figure 5-7 Unfolding the Group

Figure 5.7 is the unfolding of the group that was added in figure 5.6. The group as earlier discussed are the sensors which are regarded as birds and birds that are modelled into the system for monitoring and collection of data. From table 5.1, every sensors such as the PLCs, HMIs, temperature sensors and others have their unique ID which will be used to identify them in the system. These unique IDs can be seen in figure 5.7 as B, T, P, H, S, and A. These sensors occupy a place in the model and all these sensors are regarded as birds. However, their position can be activated whenever they are checked. The idea of the model is to check the operational activities of the sensor, to know whether they are operational or down. If they are down, their position cannot be activated, and their marking will not be available in the initial markings labelled bird. The same thing will happen in the place location, if the bird is down, their transition will not be activated, which will result in the marking being excluded or finish. In order to explain fully figure 5.7, the place scanning in figure 5.6 is the same as the place in figure 5.7. This is when the birds arrived and the ones that are scanning as well as the ones that are idling or eating. Although the ones that are idling are not included, as they are counted as dormant birds. The ones that are scanning at this time will have groups or belong to a group. These groups can be seen in figure 5.6 as Chk_Group, which is a place with an arc from the place scanning. The same can be seen in figure 5.7 as a place called group with an arc coming from the place bird. The whole process of the group formation goes on like this and many groups can be formed, and groups interact with other groups.

5.2.3 Flocking Behaviour Model

Flocking is one of the collective behaviours seen in the birds as explained in chapter 3. However, emulating such behaviours for securing Industrial Control System might be profitable especially their collective movements and detection approach which is the main

focus of this study. Figure 5.8 is the petri nets general model of the flocking behaviour seen in birds for the protection of ICS. The model shows the flock as a place in the beginning of the model. This represents the flock of birds in an environment and the transition labelled "Enter_W" stands for enter the world. A flock when present, will enable the transition for the flock to enter the world. The transition cannot be enabled if there is no token present in the place flock. When the transition fires, it will produce the neighbours, which will enable the transitions getGroup 1..n. This means that in a flock, there can be numerous groups. Birds can belong to more than one group as demonstrated in the place group #1..n. A bird in the group can either be scanning or idling and the scanning means active, while idling means deactivated or disabled. It is either a system is active and working or it is idle and silent, meaning not working. The place group #1..n will enable the transitions scan and idle if there is a token in the place. When a bird is idle according to the model, it stays in the buffer (End) and the transition cannot be activated until the counter is reached. The counter is said to be 2 within some time and the time is determined by the number of birds against the size of the system.

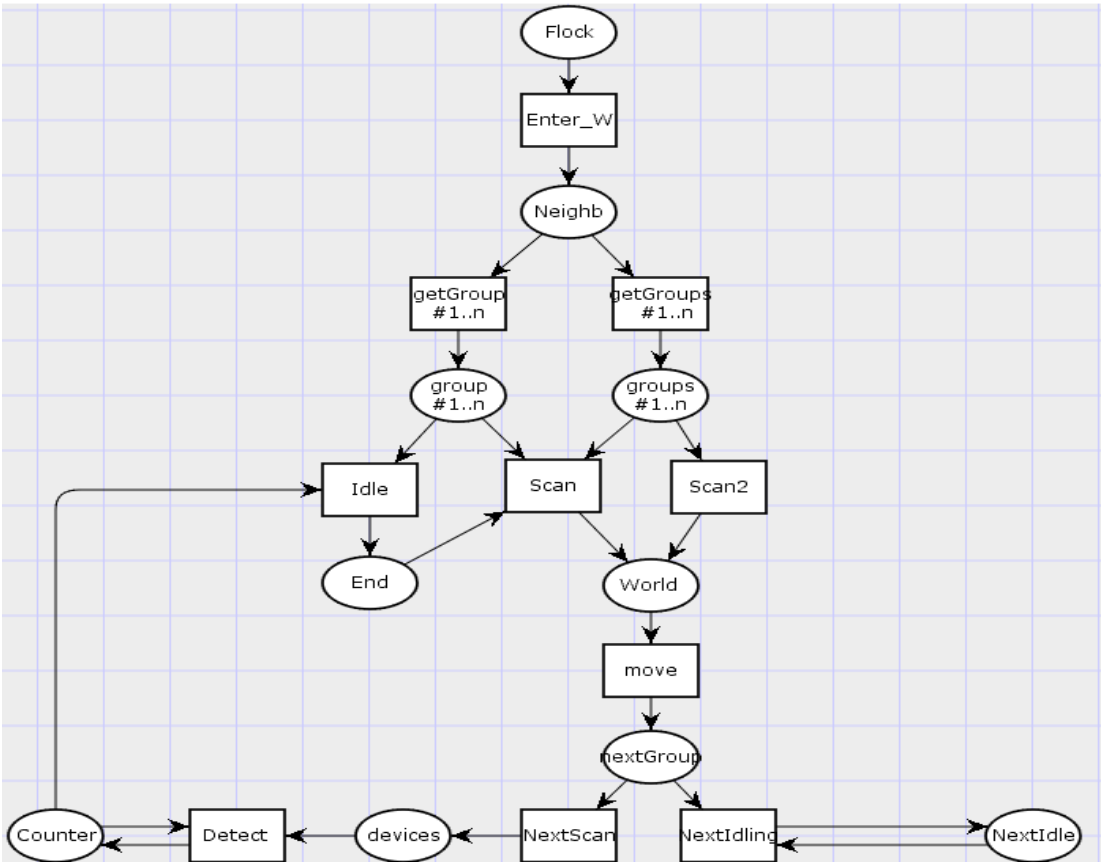


Figure 5-8 Flocking Birds Model

Every detection will be deemed very serious, but a system with up to 1000 and above sensors depending on the environment can be set to 3 detections within 5 seconds. The setting will automatically activate the idle birds which in this model means spawning. This will give rise or birth to more birds in the system as seen in the idle state with scan and scan2. It is either in the idle or it is scanning. The next will be in the world which will enable the transition "move" for the creation of the next group. This shows that a bird can move to another group or formed another group as a result of movements. The scanning and idling will continue until something is detected from the connected devices up to 2 or 3 times within 5 seconds or the set time.

There are some properties in this model that are very important for making a decision on the implementation of this model in any environment. Some of these important objects or properties as discovered in the petri nets model are as follows; Scan, Idling, getGroup, Move, Detect, Neighbours, Flock, World, Counter. More decision algorithm will be created in explaining the integration and the meaning of these objects. This will show what they are and how to incorporate them in the system in the language of choice.

5.3 Description of the Actions and the Model Functions

There are some vital element of a model that will allow others to replicate the model in their own environment. These involved some actions and functions in the diagram that needed to be understood in order to work with the model. From figure 5.1 to figure 5.8 of the models, the places and transitions have names that are vital for the model. These names or letters means something that are only associated to this model in this particular environment. The environment here is the ICS and SCADA system in particular. However, the nomenclatures need to be express in a more vivid way for anyone that would like to implement it. At different times in the model, places and transitions have been used to mean different things. In order to unite the model together for better understanding, these nomenclatures will be explained and the approach for implementation in pseudo code will be made available for the purpose of simplicity.

5.3.1 Scanning

Scanning is one of the nomenclatures used in this model to describe the active state of a bird. A bird is scanning when it is in active and alive state. Depending on the size of a system as explained above, the initial birds have infinite Time-To-Live. This means that they do not die as well as any other bird that has limited TTL. The default TTL for every other bird is from 5 to 20 seconds and this can be renewed if the group detect something in the system. Although

detection is a function on its own in this model, but it has something to do with the scan. This shows that the TTL will be renewed or refreshed to the default TTL, whenever there is a valid detection activity. The points to remember here are as follows; TTL, 5 to 20 seconds, active state or alive and renew.

Scanning
Input: b; is a bird
DA
Scan (b)

1. **While** b.TTL > 0 **Do**

   Continue looping
2. **End**

Algorithm 3 Scanning

Time-To-Live (TTL)
Initialise TTL = 20000ms
For each bird b in the world **Do**
In every iteration TTL = TTL -1
If TTL = 0
Stop the process (b)
Else
Continue
End If
End For

Algorithm 4 Time-To-Live

5.3.2 Idling

Idling defines a state where birds are not doing anything and their TTL have finished. It is an idle state where the activity is disabled, and the bird is muted, or it is not operative. It is either a bird is active in a scanning state or it is dead in an idling state. Hence, idling represent a state where a bird is not and cannot function. This state is to show that a bird can die and the dying here is that the bird (agent) is being disabled. Like in the state of scanning, a bird is either scanning or idling and as the scanning represents the active state, therefore idling is the opposite. Birds with infinite TTL do not die or goes into idling, the ones with limited TTL dies

and the length of their TTL is between 5 to 20 seconds. However, for the implementation of the idling state, the algorithm below shows the steps needed.

```
Idling
Input b; b is a bird
DA
        1. If b.TTL = 0 Do
                Discontinue (b)
        2. End if
```

Algorithm 5 Idling

5.3.3 Group in the Flock

Groups are those birds that are connected directly to a bird (Bird B) in the flock. However, a bird (Bird B) are those with the unique ID of B as seen in figure 5.1 and further simplified in figure 5.2. These groups are based on the number of neighbours that are connected to each other. Two or three birds with the unique ID starting with B can be in the same group. As stated above in section 5.2, the more the number of birds, the less the scanning frequency of each bird. Therefore, the more the number of birds with the unique ID of B are connected in a group, the less scanning for each bird. Although, this might lead to idling state of the birds with limited TTL and save memory space. The groups are determined also from the world or their environment. The fact is that the world has height and width and these measurements will help in determining the position of a bird in the group.

```
getGroup: (Return birds in that group)
Input b; b is a bird
group (b, position, cell);
#return a list of birds in a cell
Position (a tuple)
DA
Groups = []
For b in ListOfNeighbours.world Do
        get (position.b, b.ID)
        If position is within the range in the world
                count the cell
                add to the group
        End If
End For
#subject to further review as more understanding of the world structure is needed in order to
determine the position and the direction for the countina.
```

Algorithm 6 Group

In order to understand the group creation in the neighbourhood of the birds, the environment where the birds operate will be created. The environment will create a platform for understanding the group formation of the birds.

5.3.4 World

World is the environment where all the activities are virtualised. The model implements world as a grid where each bird lives in a cell and there are neighbouring cells. The information in chapter four showed clearly that the class cell has a composite relationship with the world. This means that the cell is contained in the world and without the world, there will be no cell. Each sensor in the system has a place in the world as a bird. Their location can be known through their position in the world, their cell location. Birds with unique ID of Bs are the ones that will be used to determine the neighbours and the groups as seen in table 5.2. Devices that need to be monitored have their location in the grid and their location number with their individual unique IDs, which are there for easy identification of the birds.

x/y	0	1	2	3	4	5	6	7	8	9
0	0,0 H1	1,0 **B1**	2,0 P1	3,0 S1	4,0 S2	5,0 B3	6,0 S5	7,0 T9	8,0	9,0
1	0,1 S3	1,1 S4	2,1 B2	3,1 **H2**	4,1 S3	5,1 S4	6,1 B4	7,1 T8	8,1	9,1
8,2	0,2	1,2 T4	2,2 P2	3,2 B5	4,2 P3	5,2	6,2	7,2	8,2	9,2
8,3	0,3	1,3 H3	2,3 B6	3,3 H4	4,3 H15	5,3 S21	6,3	7,3	8,3	9,3
4	0,4	1,4 T9	2,4 T8	3,4 A13	4,4 B7	5,4 S20	6,4	7,4	8,4	9,4
5	0,5	1,5 S6	2,5 B8	3,5 S7	4,5 P5	5,5 B10	6,5 P6	7,5	8,5	9,5
6	0,6	1,6 S9	2,6 S10	3,6 A1	4,6 B9	5,6 A2	6,6	7,6	8,6	9,6
7	0,7	1,7 A3	2,7 B12	3,7 P8	4,7 S12	5,7 B11	6,7 S13	7,7	8,7	9,7
8	0,8	1,8	2,8 A8	3,8 B13	4,8 A9	5,8 S14	6,8 A10	7,8 B14	8,8 T1	9,8
9	0,9	1,9 T2	2,9 B15	3,9 T3	4,9	5,9 A11	6,9 B16	7,9 P8	8,9 H9	9,9

Table 5-2 10x10 World Grid

```
World (This is the environment where the birds' live)
Initialise GridWorld
#Implement a grid in the world based on the row and column
Int Grid = New Grid (10, 10)
   For (int i = 0; I < 10; i++)
         For (int j = 0; j<10; j++)
               Put birds in the grid (i, j), based on the location
         End for
   End for
End
```

Algorithm 7 World

Table 5.2 is a simple grid world that was created for this model in order to further simplify the understanding of the model. In the upper left corner of the grid are six highlighted cells of the grid. In the six of the cells are two cells with B1 and B2. These two Bs are the birds that connect other birds in the group. The six cells formed a group with the first bird, the B1 as the one that was used to locate the group members. Such type of bird can have other bird like them in the same group as seen in the sets below. This will give an insight into how the connection between birds and the devices will flow in their world.

B1 = {H1, S3, S4, B2, P1}
B2 = {H2, S1, P1, B1, S4}
B3 = {S1, S3, S4, B4, S5}
B4 = {T8, T9 S5, B3, S4}
B5 = {P3, S3, H2, B2, P2}
B6 = {H4, B5, P2, T4, H3}
B7 = {S20, S21, H15, H4, A13}
B8 = {S7, A13, T8, T9, S6}

The world or the environment here is the beginning of the whole algorithm and getting it right is paramount for this project. The world will involve all the activities as seen in table 5.2 which has all the values as the content of the table. The table represents the grid world with all the values in the world such as the sensors and birds represented with their unique IDs. The sensors are birds themselves, but their actions are deterministic, and they do not move. The other birds with the unique IDs starting with B are stochastics and moves around the world. They have states such as infinite TTL and limited TTL, they can be in different locations and have different neighbours and belong to different groups. The view of all the requirements and capturing all that is needed, Markov Decision Process (MDP) will be a better option for this. Hence, MDP will be partially looked into for a better understanding of the project.

1. States: based on the information in table 5.2, states started with 0,0 to 9,9. This is just a small sample of 10x10 prototype grid world.

2. Actions: actions here represent the move, getting the number of groups, finding the nearest neighbours and updating the TTL.
3. Transitions: T (s, a, s') which represents the state, action and the new state.
4. Rewards: this represents the state when something is detected, and the reward is increase in the TTL and when two groups detects, the reward will be global increase in TTL. When TTL finishes and the bird is deactivated and removed.
5. Discount factor: determines the value of future reward to a bird in the environment.

5.3.4.1 Determining the State and Action

Determining the state and action is not about the whole birds since the sensors are static while the movement of birds with the unique ID starting with B are dynamic. Hence the focus is on the birds with the dynamic state, their position in the world and their operations.

S = {1.0, 5.0, 2.1, 6.1, 3.2, 2.3, 4.4, 2.5, 5.5, 4.6, 2.7, 5.7, 3.8, 7.8, 2.9, 6.9}

Here are 16 set of state that are being occupied by 16 birds in their world. These states can be bigger as well as the birds depending on the size of the world. The model is using 10x10 grid world for the birds. The other sensor birds are deterministic, and their condition is either dead or alive. This information is received by the bird that come into the environment showing that the neighbour or group members has the status of either 1 or 0. As the S above represents the states of the birds and each bird in the world has 15 possible state it can attain at every move. When a bird moves into another state, it receives reward which are its neighbours in this case. The neighbours form its group and their status will show whether they are dead or alive. One thing that should be clear here is the initial placement of the birds in the world are. Birds are placed in certain state based on the neighbours and groups such as sensors and other hardware that are regarded as birds. However, the sensors and other hardware regarded as birds do not move, only the birds (with B ID) moves and their movement is non-deterministic. The model for the world in this project will be characterised by these 5 quantities such as action, transition probability, reward and discount factor as explained above. These can be represented as 4 tuples (S, A, T, R) where S is for state, A is for action, T is for transition, R for reward. Since S has been stated above and A is a finite set of actions, the rest of the quantities are as follow;

- $T(s'|s, a)$. This is called transition probability which is the probability of a state 's' with action 'a' arriving at a new state s'

- $R(r|s)$. This is the probability of receiving reward when arrived in state 's'. The values that accompanies this are either positive or negative. Hence, negative means deduction while positive means addition.
- $\gamma \in [0,1]$. This represents the difference in value between the future and presents reward.

5.3.4.2 Determining the Value Functions and Policies

The value and policies are simply to outline the quantities that will help in structuring the behaviours of the birds in the environment. Where and how the birds are expected to behave and what they will be expecting under certain circumstances. Below are the two quantities;

1. Value function which is represented as follow; Q^{π}(s, a). It represents different action 'a' carried out in a state 'a' under the policy π that will result in the array of Q. This means that the action is based on the state and value.
2. Policy is another quantity that is represented as $\pi(a|s)$. It is the probability of action 'a' when in a state 's'. Some of the birds have a specific action and they are deterministic while some are not deterministic. The ones that are deterministic are based on the value of the state, it is called state value function which is represented as follow; V^{π}(s). This implies that an action happens as a result of the policy for the state a = π(s).

Table 5.2 has 16 possible dynamic states or states that are not deterministic and 45 states that are deterministic. Hence the reward is mostly for the state that are not deterministic without probability. The reward for all the birds in this case are the same, which is their neighbours and any detection will increase their TTL as a group. Hence reward is still based on the state and action. The state will be the new state and action will lead to new action that will either increase the TTL or remain the same. Now TTL increase is a reward for the action 'a' in a state 's'. However, since the sates are finite, there is no need of considering infinite states for the reward. This will be represented as follows;

$$r16(s) = r(s) + r(s_{1.0}) + r(s_{5.0}) + r(s_{2.1}) + r(s_{6.1}) + r(s_{3.2}) + r(s_{2.3}) + \dots r(s_{6.9}) \quad (1)$$

The reward for any move or new state is the number of neighbours present in the group based on the new state. Hence when in state 's' and take action 'a' based on the policy π. This can be represented as follow; $E\{r(S)\}_{\pi}$. Therefore when according to the policy π from state S, the total discounted payoff which is referred to as value function for the policy π is as follow;

$$Q^{\pi}(s,a) = E\{r(s) + \gamma r(s_{1.0}) + \gamma^2 r(s_{5.0}) + \gamma^3 r(s_{2.1}) + \gamma^4 r(s_{6.1}) + \gamma^5 r(s_{3.2}) \dots\}_{\pi} \quad (2)$$

How to know the reward and the value for the state have been known, but the neighbours or the group which is one of the rewards is not known yet. Therefore, knowing how to determine the groups will also be of utmost important than any other solution here.

5.3.4.3 Determining Group Members

Determining the members of a group of birds, the current position of the bird in the cell should be known. Bird B1 location in the world is 0.1 and its neighbours and group members are {H1, S3, S4, B2, P1. The location 0.1 is an indication that there is a neighbour before that in the location 0.0 which is one of the neighbours to location 0.1. The world is a 10x10 grid world and it has Hight and width. These locations are the cells in the world. For this model, cells were used for the identification of the location as well as the neighbours as seen in table 5.2. There are six important rules to consider here;

1. When a bird with unique ID starting with B is in location starting with 0 such as bird B1.
2. When a bird with unique ID starting with B is in a location starting with other numbers that are not 0 and immediately followed by 0. Example is if a bird is in location 2.0, 3.0, 4.0, 5.0…n.0
3. When a bird with the unique ID starting with B is in location 0.0.
4. When a bird with the unique ID starting with B is at the other end of the environment such as location 9.1, 9.2, 9.3, 9.4, 9.5, 9.6…9.9 as the grid world is 10x10
5. When a bird with the unique ID starting with B is in location 9.0.
6. When a bird with the unique ID starting with B is in any other location apart from the ones mentioned above.

These are six important rules that will govern the location of neighbours or group in the world, named Individual and Group Location (IGL). The first five are the rules that will help in locating groups when a bird is located at the boarder of the world, while number six will be for every other position in the world. Figure 5.10 a-f will be a demonstration of the approach. This will show how different bird's locations in the world will be determined based on the IGL. Figure 5.9 is a graphical representation of the positions within the x and y axis of a graph. This will aid in easy identification of groups within the grid world. Hence, this model used x and y two-dimensional world that can locate any bird in the grid as seen in figure 5.9 and demonstrated in figures 5.10 a-f

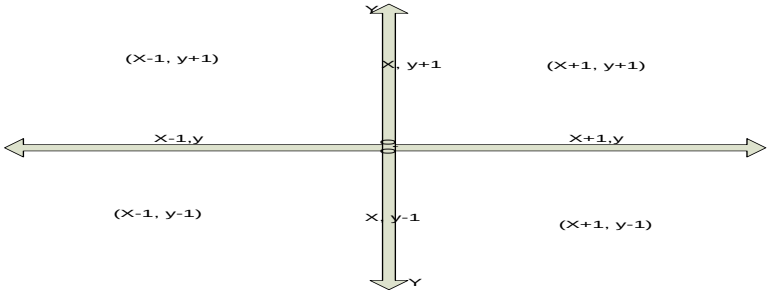


Figure 5-9 Position in the World

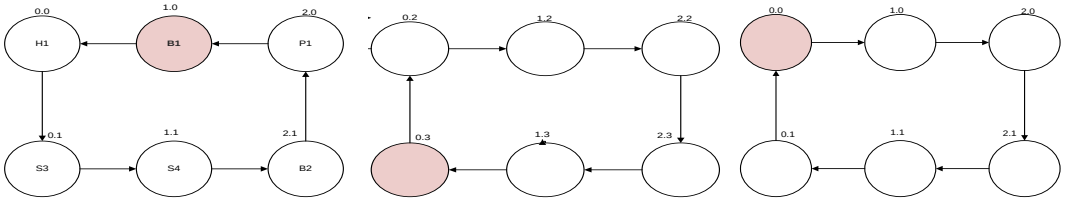


Figure 5-10e 5.10a (1) *Figure 5.10b (2)* *Figure 5.10c (3)*

Determining the neighbours of the Bird B1 **Direction is determined from 0.3 position** **Direction is determined from 0.0 location**

B1 = (x, y);
H1 = (x-1, y) path: B1 -> ()
S3 = (x-1, y-1) path: H1---> (x-1, y-1)
S4 = (x, y-1,) path: S3 -> (x-1, y-1, x+1)
B2 = (x+1, y-1) path S4 (-> (x-1, y-1, x+1, x+1)
P1 = (x+1, y) path B2-> (-> (x-1, y-1, x+1, x+1, y+1)

For 5.10a

The shaded position 0.3 is p = (x-1, y)
P 0.2 from 0.3 = (x-1, y+1)
P 1.2 = (x, y+1)
P 2.2 = (x+1, y+1)
P 2.3 = (x+1, y)
P1.3 = (x, y)

for 5.10b

The shaded position 0.0 is p = (x-1, y)
P 1.0 = (x, y)
P 2.0 = (x+1, y)
P 2.1 = (x+1, y-1)
P 1.1 = (x, y-1)
P 0.1 = (x-1, y-1)

For 5.10c

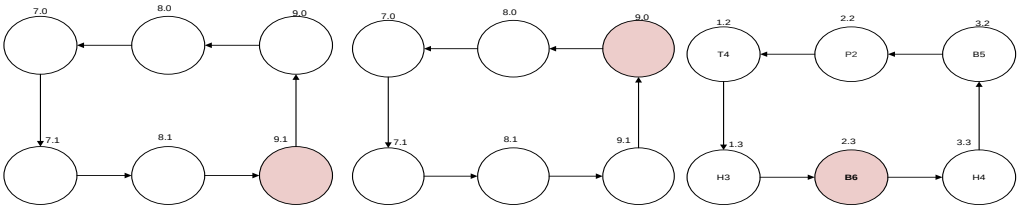


Figure 5.10d (4) *Figure 5.10e (5)* *Figure 5.10f (6)*

The shaded position 9.1 is p = (x+1, y)
P 9.0 = (x+1, y+1) path:
P 8.0 = (x, y+1) path:
P 7.0 = (x-1, y+1,) path:
7.1 = (x-1, y) path
8.1 = (x, y) path

For 5.10d

The shaded position 9.0 is p = (x+1, y)
P 8.0 = (x, y)
P 7.0 = (x-1, y)
P 7.1 = (x-1, y-1)
P 8.1 = (x, y-1)
P 9.1 = (x+1, y-1)

For 5.10e

The shaded position B7 = (x, y)
H4 = (x+1, y)
B5 = (x+1, y+1)
P2 = (x, y+1)
T4 = (x-1, y+1)
H3 = (x-1, y)

For 5.10f

Figures 5.10 (a-f) are the representation of the IGL rules that will govern the location of groups in the model. Birds can be anywhere in the world such as in the, middle, boarders, front and back. In the grid world represented in table 5.2 are these six locations. The question now is how to represent this in equation. The boxes bellow the figures 5.10 a-f are the simplification of how to represents the groups. In order to represent and better explain it, important values must be identified by following the idea of Markov Decision Process (MDP) for value identification. The two most important factors here are the location and direction. The location here is the state and the direction is the action. In order to find the action or series of action that will lead to the right place which is state. Identifying the right policy for the group action will be the next step which is; $\pi = (s, a)$. π represents the right policy for the state and action. The action here are $A = (East, West, North\ South)$. Hence ($a \in A$), meaning that 'a' is an element of A or 'a' is in A. Based on the information in figures 5.9 and 5.10 a-f, x and y axis was used in determining the location on the grid.

1. Figure 5.10a showed that the bird is located in position 1.0 in the grid world. Here the optimal policy showed that when a bird B is in a location with y = 0, the first direction is towards east. $\pi_s^* = \underset{a \in A(s)}{\arg max} \sum_{s'} P(s'|s, a)\,(U, s')$ (3)

   This means that State $S_{xy=\,a}$, $x > 0\ and\ y = 0$. This define the case where y must be equal to 0 and x greater than 0 and all other possibilities as listed in section 5.3.4.3. Hence the group will be located as follow;

```
Get bird. b (grid, x, y)
Parameters:
        grid: two-dimensional world of cells
        x: the row of the cell
        y: the column of the cell
Return
        The bird b in the cell (x, y)
X = Len (grid)
Y = Len (grid[0])
For bird b in the grid (x, y)
        If x >0 and y = 0
                Follow steps in figure 5.10a
                Return groups
        Else if x = 0 and y > 0
```

Follow steps in figure 5.10b

Else if x = and y = 0

Follow steps in figure 5.10c

Else if x. Len. Count -1 and y > 0

Follow steps in figure 5.10d

Else if x**.** Len. Count -1 and y = 0

Follow steps in figure 5.10e

Else

Follow steps in figure 5.10f

Return None

end if

end for

Algorithm 8 Algorithm for the Position of Bird in the Cell

Before delving into other part of the model, the environment or the world has been defined in section 5.3.4 based on some characteristics and properties available. It is a grid world with cells, where birds live in the cell and have their neighbours and groups. The grid world has boundaries and sizes as illustrated above in figure 5.10 a-f. This determines the location of neighbours and group in the grid. The reward for every new bird that is added to the cell is its group members. Now the next decision is on the movement of the birds in the grid. What determines or influences their movement? Do they move based on the time that was set and what is the time if that is the case?

In the 10x10 grid above on table 5.2, there are sixteen bird's B with their unique IDs. These birds are in different locations in the grid world. Their movements are random, and mostly within the cells as they exchange cells between each other. There are other cells that have other types of birds such as HMI, PLC Actuators and among others. These types of birds are static, they do not move compared to the ones with the unique ID starting with B. The ones with ID starting with B moves but within themselves. This is to avoid situation where the neighbours will be left without this type of bird. These are the birds with infinite Time-To-Live (TTL) and they do not die or become deactivated because their TTL does not finish. The second type of bird are the birds with unique ID other than the ones mentioned above. These types of birds can die and can also come back to life or be born (death and birth). These birds will occupy the other empty spaces in the grid and move around between cells as far as it is empty and their TTL is still valid. They have TTL of some seconds depending on the environment. They die, or the process is deactivated when their TTL finished and come back to live when two groups in the grid detect something bad or anomalies within their TTL. The birds are to help in the

network expansion of the system when necessary. If more devices are to be added to the system, some of the birds that are connected to them will receive infinite TTL. Hence below is the decision and definition of the movement among the birds with infinite TTL.

$S_{16} = (s_{1.0}) + (s_{5.0}) + (s_{2.1}) + (s_{6.1}) + (s_{3.2}) + (s_{2.3}) + \dots (s_{6.9})$

These are the 16 states the 16 birds are living in based on the information in table 5.2. In these locations are birds, and they have to move at a time step, each to another state or location. Determining the process of moving among these birds is the next step and they can only move in these 16 locations based on the grid above in table 5.2.

$U([s_{1.0,}s_{5.0,}s_{2.1,\dots,}s_T]) = \sum_{t=0}^{T} R(s_t)$

The sequence of the state here is from 0 to T which amounts to 15 states. This is based on the model and the states that are being modelled have 16 states (see table 5.2). The optimal policy for the states can be written as follow;

$$\pi_S^* = \arg\max_{\pi} U^{\pi}(S) = \pi_{s'}^* \; for\ all\ new\ state\ s' \dots\dots \quad (4)$$

The equation 4 above is saying that the optimal policy from state 's' is the same as the s' which is the new state. With this we can determine the optimal policy of the bird's movement to other cells based on the reward.

Move
Bird b has a state 's' and action 'a' that will result in new state s'
s + a = s' = s'_{15} ##one bird has 15 possible locations or states at every move action in the environment
current state = s
Current Location = s_a
move = $s\ a\ \rightarrow\ s'$
DA
Initial states (s) = 0
For b in location $s_{a\dots s_n}$ **Do**
    check for a valid next state in (east, west, north, south)
    take the nearest empty state
    move to new location (b)
    delete from the old location (b)
end for

Algorithm 9 Move

In order to generally understand this decision for the 16 states available in this model, the Markov Decision Process will be drawn. The 16 states will be represented in the model that will show how they will likely move from one state to the other between themselves. This movement is based on the optimal value as seen in figure 5.11 and table 5.3.

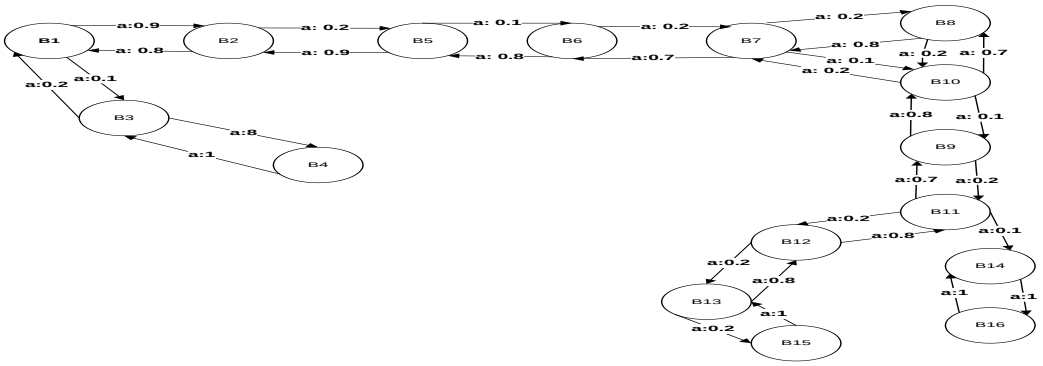


Figure 5-11 Decision Process for the 16 States

The transition function for the state as seen in figure 5.11 is as follows $T(s, a\ s^{'}) = p$

s	a	p	s'
B1	a	0.9	B2
B1	a	0.1	B3
B2	a	0.2	B5
B2	a	0.8	B1
B3	a	0.8	B4
B3	a	0.2	B1
B4	a	1	B3
B5	a	0.9	B2
B5	a	0.1	B6
B6	a	0.8	B5
B6	a	0.2	B7
B7	a	0.7	B6
B7	a	0.2	B8
B7	a	0.1	B10

Continuation of table 5.3

s	a	p	s'
B8	a	0.2	B7
B8	a	0.8	B10
B10	a	0.7	B8
B10	a	0.2	B7
B10	a	0.1	B9
B9	a	0.8	B10
B9	a	0.2	B11
B11	a	0.7	B9
B11	a	0.2	B12
B11	a	0.1	B14
B12	a	0.8	B11
B12	a	0.2	B13
B13	a	0.8	B12
B13	a	0.2	B15
B15	a	1	B13
B14	a	1	B16
B16	a	1	B14

Table 5-3 Transition Function

5.3.5 Flocking

Flocking here is a form of interaction between birds as well as their movement in the world. The flocking will define the number of birds that will interact with each other as well as the environment where they live. It will show how information travel within the flocks. Table 5.3

has shown the transition function of the birds as seen in figure 5.11. This represents the transitioning and possible movements of the birds in this model. It will also help in determining their interaction based on their state. If the state is known, the neighbours are known, interaction can be determined.

State ‘s’ of bird ‘b’

A is a set of actions for the bird’s B in the neighbourhood. This can be represented as follows;

A = $(a_1, a_2, a_3, a_4, \ldots a_n)$

Flocking
Declare Number of Birds B
Create a mailbox for the collection of massages
Alive = 1
Dead = 0 (no response)
Action a_1 = broadcast a_1
Action a_2 = report neighbour with 0 response
Action a_3 = clear the mailbox
DA
Begin
    For bird b in listOfBirds **Do**
        For every move from $s \rightarrow s'$
            Perform ($a_1, mailbox$)
            clear after 2 mins (a_3)
            if response = 0
                show (on screen)
            End if
        End for
    End for
End

Algorithm 10 Flocking

5.3.5.1 Communication Between Birds in the Flock

Birds with the unique IDs starting with B do move from one designated location to another as seen in figure 5.11 and table 5.3. They can only move between themselves in the flock. Once there is a move, their previous location will be deleted, and they will be in a new location. In this new location, they will search for their neighbours and try to complete the handshake. If the handshake is complete, the neighbour is alive but when the handshake is not complete, the neighbour is dead, and a retrial is needed. HMIs, PLCs and their various protocols constitutes neighbours as well as birds with the unique ID B. These neighbours have their own unique IDs that identify them as seen in table 5.1 but do not move. Those with other IDs apart from B are

static and can only communicate with the bird with the IDs starting with B in their group. Below is the algorithm pseudocode for this section of the model.

PLC, HMI SCANNING
Bird with the unique ID starting with B is in new location

$$s + a = s' = bird\ B$$

Connected = 1
Not connected = 0
Search the group
If the condition is true *#bird B in a new location or cell*
    Triger the get group action
    Try connecting to PLCs/HMIs
        If the connection was established
            Return 1
            Save the message with the ID on mailbox
        Else return 0
        Check for timeout on the process
            If connection was not established and timed out
                Return 0
                Retry connection one more time
            End if
        End if
End if

Algorithm 11 PLC/HMI Scanning

5.3.6 Detection

Detection is the vital part of the collective behaviour modelling of the birds. These bird's functions in their environment as modelled above in section 5.3.4. They communicate with their environment as well as with other birds in their group. They check among themselves for any anomalies in the system, which this model identifies as the predator. Birds with the ID starting with B are the mobile birds while PLC and HMI as well as other sensors are not mobile. Therefore, the ones with the mobile capabilities query others in their group for their status. If a sensor during querying does not respond or complete the handshake, it would be deemed as an anomaly and would be reported to the operator for further investigation. The operator has the table of all the sensors and their location as seen in table 5.1. However, the model here detects when a system is down and reports it to the operator to avoid further disruption of the system. Other types of anomalies can be introduced and modelled such as buffer overflow, Denial of Service (DoS) and among others. This is an introduction into other things that can be modelled into this approach. One and vital thing is that the approach is viable and applicable

to other systems. Figure 5.12 summarises the approach of detection in this model where bird detect anomaly, reports it and move to the next state.

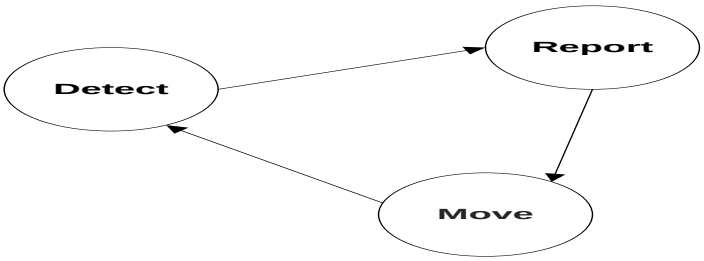


Figure 5-12 Decision Process for the 16 States

Detection
Bird with the unique ID starting with B is in new location

$$s + a = s' = bird\ B$$

Connected = 1
Not connected = 0
For bird B in the ListOfNeighbors **Do**
If the condition is true *#bird B in a new location or cell*
Triger the get group ()
Try connecting to PLCs/HMIs
If the connection was established
Return 1
Save the message with the ID on mailbox
Check the input from HMI to PLC
If the input will result in unreliability
Report it ()
End if
Else return 0
Report it ()
Check for timeout on the process
If connection was not established and timed out
Return 0
Retry connection one more time
Report it ()
End if
End if
End if
End For

Algorithm 12 Detection

Summary

The model in this chapter has proven the viability and the possibility of the application of detection approach seen in the flocks of birds for anomaly detection. It proves that every movement and behaviour can be modelled and apply for the benefit of securing ICS. The benefit of group living and the application of it in detection was highlighted. This idea was captured in figure 5.1 and in the entire chapter. This demonstrated the free coloration of information within the flock and groups. Therefore, information travels from one bird to the entire flock. The model demonstrated and showcase some important element and properties seen in the flock of birds. It shows the movement of birds from one group to another. The model demonstrated the important of information transfer from birds in section 5.1.2. The information travelled from a single bird to its group and the entire flock. Hence, a bird is either scanning or idling and the scanning in this model demonstrates the active state of a bird while idling is the inactive state. Birds with the IDs starting with B can move to a different location, but they have infinite TTL. Sensors in this model are categorised as bird but they are static and do not die also. Any other birds have limited TTL that is based on the definition of the operator. In this initial demonstration, the TTL is set to 20 seconds which is 20000ms.

The model was further expanded and demonstrated by modelling with Petri Nets formalism. Petri Nets was used to model a single bird that is either scanning or idling as well as groups and their location in the flock. The location in this model was modelled as the state of the bird, because the model is made of a grid world. The world has cells that can only accommodate one bird per cell in cell. This is a maximum of 6 birds in a cell. In the petri nets model, birds can be added as well as being removed in the group. The removal is as a result of TTL finished and addition is a process of giving birth or spawning as described in the model. Whether groups formation or individual bird's location change, addition and removal as well, are decisions that has to be modelled.

Decision algorithm was modelled and designed for the model in the area of group formation, location change and movement as well as detection and among others. This identifies the condition for movement, detection and change of state and group formation. The model expresses the collective behaviours seen in the flocks of birds. These behaviours have been emulated for the detection of anomalies on ICS. Hence the whole approach is summarised in three steps which are Detect, Report and Move (DRM).

CHAPTER VI

6 ANALYSIS OF THE MODEL ON THE CASE STUDY

Chapter four and chapter five delved a little bit into the case study in explaining all that is needed to setup the case study scenarios. The case study in this project is the train system that was explained in the previous chapters. The reason for chosen the system was also justified in chapter four of this thesis. The reason being that it was the available platform for this project. Hence most of the required material and devices for the SCADA system are built into it.

The train system that was defined in chapter four has two siemens PLCs and 1 Schneider electric PLC. The track points which are connected to the PLCs are the sensors that controls the movement of the train or locomotive. They switch the train direction when operated or when command is issue from the HMI to the PLC. Based on the information gathered during data collection in chapter four, the track point sensors in this case study has only one data type. The data type is based on the state of the sensor and the sensor can only be in two states; ON or OFF. It is either 0 or 1 which is a Boolean representing true or false state. Therefore, the sensors can either be open which is true or close which is false. Based on the analysis of the data collected, when a sensor is in a false state, it means that the train cannot go in that direction. When in a true state, train can come in or use that track to other destinations.

This chapter will start by modelling the train system tracks in order to capture all the required information. The tracks will be modelled first with the sensors and all the required points on the system. These points will determine the location of the train or it will help in identifying train location. Hence, many points are needed on the track in order to capture the movement of the train and be able to model other activities such as security. On completion of the track modelling and the points as well as the endpoints and the crossing, train will be added. The case study has two trains and their movement will be modelled to be clockwise and anticlockwise. One will be clockwise and the second one will be anticlockwise. The behaviour will help in playing up some of the discovered risk scenarios such as collision and deadlock. After the trains have been modelled, the analysis of the system will be performed for deadlock and collision anticipations. This will help in determining the viability of the system that was modelled. The introduction of the flocks of birds' model into the system will be the last for the monitoring of the so-called risk areas that was mapped out.

6.1 Case Study Train Track Modelling

The train track or the rail system in figure 6.1 is the model train that is used for this project. The rail system has 10 track points labelled TP. The track point is the switching system that will direct the train as indicated by the arrows. The track points show that trains can travel both clockwise and anticlockwise. The train track is about 10 meters with two outer tracks and two inner sections as seen in figure 6.1. The track is further divided into 10 sections from 1-10. The rail system is made up three PLCs and two HMI that are connected through network switch. The track points are connected to the PLCs through the PLCs output points. The PLCs receives command from the HMIs and execute those instructions which is evidence in the opening and closing of the track points. The engineering workstation is connected through the network switch as well as the PLCs and HMIs. The model and analysis of the case study will follow a step by step procedure, which is as follows;

1. The track system will be modelled, and the track points will be marked
2. Analysis of the track system
3. Train will be added for further analysis (2 -3 trains)
4. Flocks of birds will be added into the system for the detection of anomalies

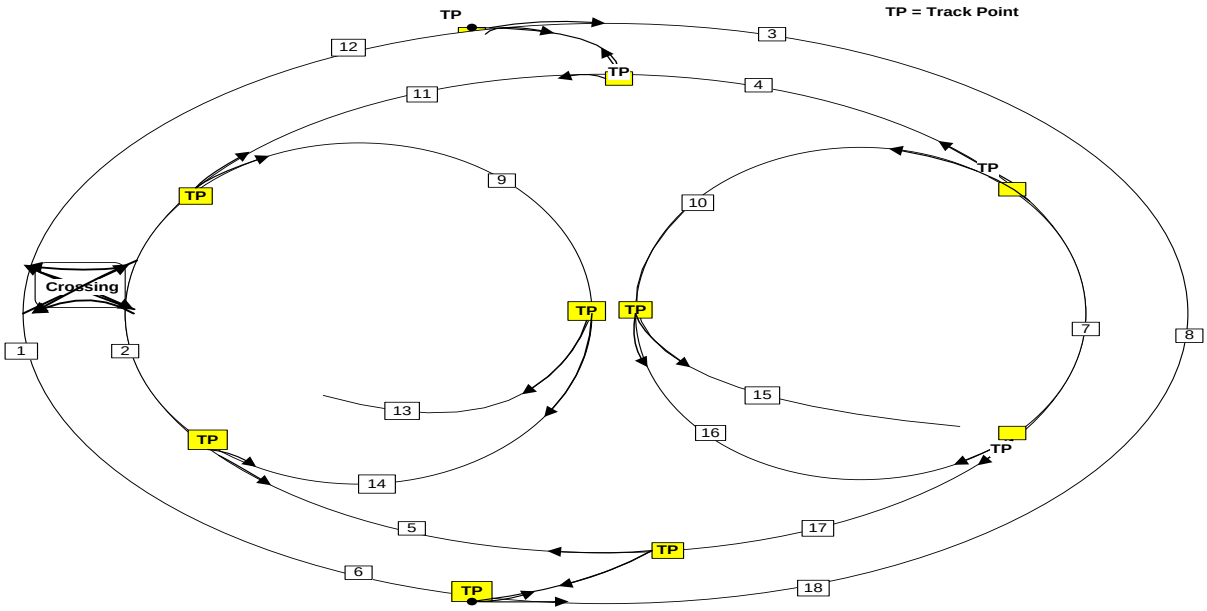


Figure 6-1 Case Study Train Tracks

6.2 Railway System Tracks

Figure 6.1 is the track model of the railway system for this project. The railway track is made up of 10 track point sensors or switches named TP which is short for track point as seen in figure 6.1. The switches are for directing the train in the track. As earlier explained, trains can move in both clockwise and anticlockwise direction. Places are labelled from 1 – 18 which is for the locations of train before and after the switches. These locations were further scrutinised by modelling it using petri nets formalism. In petri net, some locations were not counted as they are not beneficial at this time. Location numbers 1,8,12,18 were not included in order to make the model smaller since the whole information can be captured in this form. Figure 6.2 is the model train track with all the places and transitions that are needed. However, by further analysis of the train, figure 6.2b was modelled to include extra information that will help in avoiding some obstacles along the line. Buffers were added to the train track as seen in figure 6.2b below.

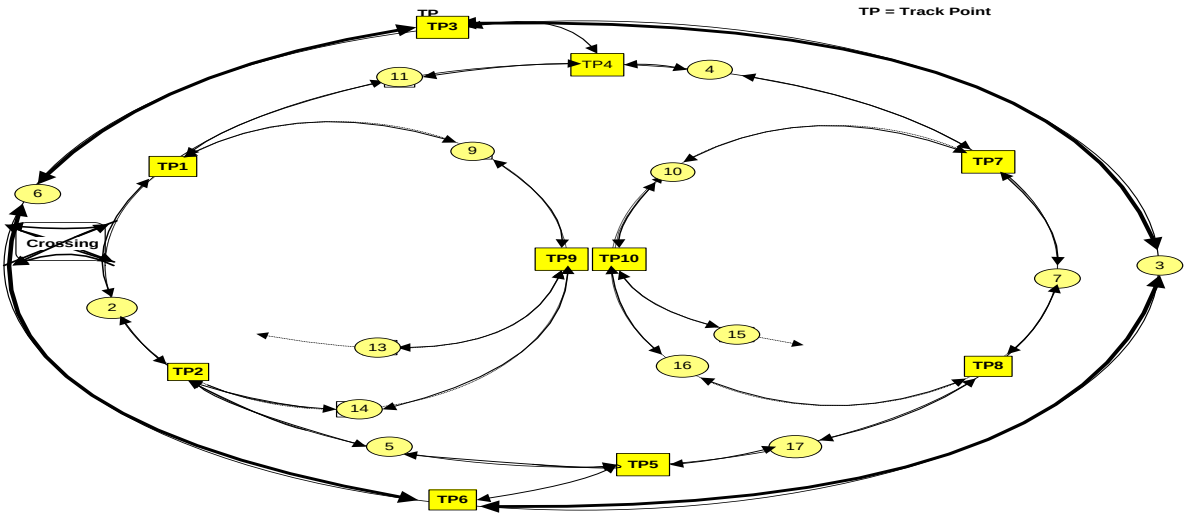


Figure 6-2 Train Tracks with Transitions and Places

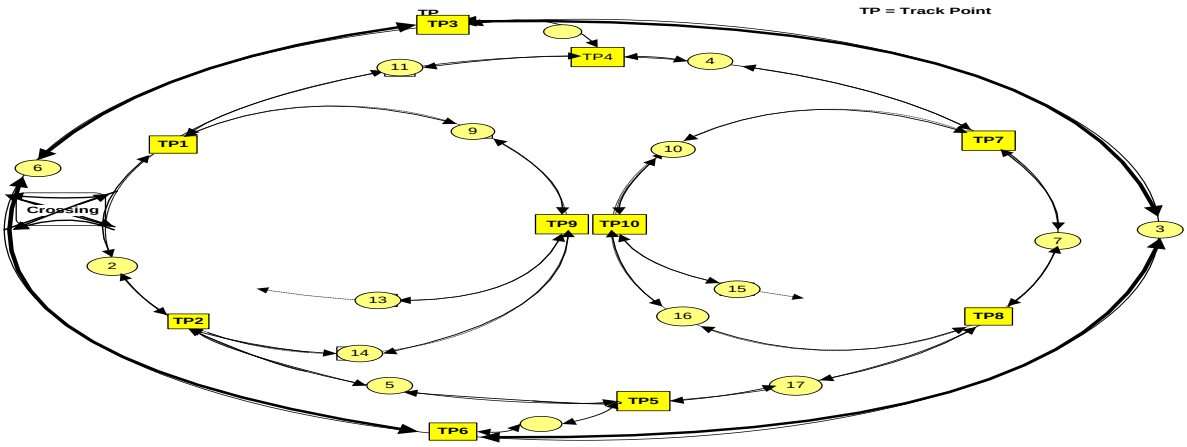


Figure 6-2b Train Tracks with buffer between TP5 and TP6, TP4 and TP3

In figure 6.2, the track points were turned into transition as it is the decision point for the train movements. Places are now integrated into the track system in order to track down the movements of the train as well as their locations. These places will help in the analysis of the train movement later in this chapter. The TPs are numbered from 1 to 10 for the purpose of analysis. The analysis will determine whether there is going to be a deadlock in the system and for mapping out the locations that are likely to cause collision. Through the analysis of the railway track, some vital information that will be fed into the model birds will be discovered further. The information such as when to stop the train and to foretell and anticipate the possibility of an incident occurring in the system. whether the train is moving clockwise or anticlockwise, with the information collected during the analysis, it will be easy to detect such an incident before it occurs in the system.

6.2.1 TP1 for Transition 1

The transition TP1 has access to places number 2, 9, 11. These places allows both clockwise and anticlockwise movements of the train based on the direction of the arrows. The arrows that goes into the transition are in both ways, therefore the train can be in clockwise as well as anticlockwise direction. If TP1 is open, trains from place 2 will follow the direction to place number 9 until it gets to another transition which is transition TP9. However, if the TP1 is closed, the direction will be to place number 11 from place number 2 and further to TP4.

Figure 6.3 will produce a deadlock if the train is in place 11 heading anticlockwise, place 9 in the direction of anticlockwise and place 2 in clockwise direction. The trains will all meet at the track point TP1 and will result in a deadlock. The TP1 has two state which are either open or close state. When the TP1 is in open state and the three trains are coming as explained above, it will surely result in a deadlock. In this condition, it does not matter whether the state is open or close, the fact is that the train is coming from the three places and the open means that train from place 2 can enter place 9. However, close means that the train from place 2 cannot stop or follow the direction of place 9 rather, it will follow to place 11. The case of allowing three trains on the same track in opposite direction is an anomaly in the system, therefore should be avoided. Figure 6.3 showed clearly the anomaly this posed to the system and figure 6.4 provided a solution to this anomaly.

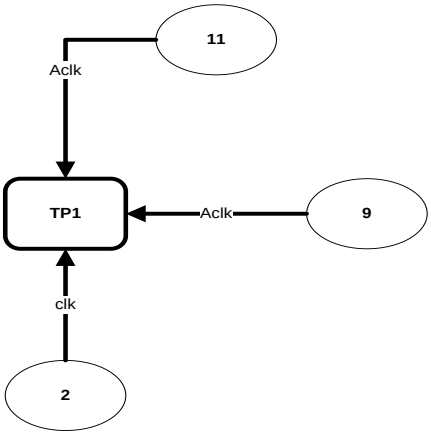


Figure 6-3 TP1 with three incoming Trains

In a situation as seen in figure 6.3, the solution can either be to avoid it happening at all or to provide a wait in form of a queue or diversion for the trains. The deadlock condition will be detected through the communication mechanism among the birds in the groups and inter groups. When the transition TP5 is disabled or closed and train is coming from place 17 to place 5 heading to 2 in a clockwise direction, TP4 should be open to allow train in anticlockwise direction to take the outer route for collision avoidance as well as deadlock. when a train is at location 2 heading towards TP1 in clockwise direction and another train in location 13 or 14 heading towards TP9 in anticlockwise direction, there is a possibility of collision. The solution to this will be to apply either a wait at TP9 for the trains at location 13 and 14, by disabling the transition and allowing the train at location 2 to proceed with TP1 closed. These scenarios can be approached for a solution as follows;

1. When a train is in location 13 or 14 heading towards TP9.
    - Location 2 should be checked to make sure there is no train heading toward TP1
    - If there is a train in 2 and in the clockwise (clk) direction, TP1 should be closed which the next location will be 11
    - The two trains should be timed to avoid collision or deadlock
2. If a train is in location 2 and heading to TP1 and another train is in location 9.
    - Delay should be applied to the train in location 9 to allow train in location 2 access
    - Transition TP1 should be closed to avoid deadlock and collision
3. If a train is in location 11 and heading towards TP1 and another train is in location 2 and heading towards TP1.
    - The transition TP1 should be enabled to allow the train in location 2 to enter into location 9 and turned back after some delay in location 9.

- Train in location 11 should be delayed allowing the train in location 2 to move first.
- Transition TP2 should be closed to avoid the train in location 9 entering location 14 after location 2
- When the train from location 2 enters location 9, it should be delayed until the train from location 11 gets to location 5.

4. There should never be a condition where train will be in location 2, 9 and 11 as seen in figure 6.3. This should be avoided as follows;
    - If a train is in location 4 heading towards TP4, a check should be applied for location 13, 14 and 9 for anticlockwise direction. If there is, the transition TP9 should be disabled or TP4 should be enabled or open to allow the train in location 4 to head to TP3.
    - If a train is location 17 and the transition TP5 is disabled or closed and a train has been detected in location 4 or 11 and 9 in anticlockwise direction, the TP5 should be enabled and there should be a delay for the train in location 9 to allow the one from location 11 go first. The train in location 17 will follow the transition TP6 to avoid collision with trains from locations 11 and 9.

6.2.1.1 When a train is in location 13 or 14 heading towards TP9

Based on the information in section 6.2.1, some important properties have been captured and documented that will aid in modelling this approach. The information supports processing of one job at a time. These shows that only one train is allowed access at a time. This will facilitate timing and delays for the train system. Two trains cannot turn up at a station, hence collision and deadlock will be avoided. Modelling this will need a place for processing the train and timing them. Two output locations for the train in location 2 and train in location 13 or 14. The processing location will be called process with arc inscription pro. The place process will have one token that will help in enabling the transition TP1. The initial location contains three trains; Loc2, Loc13 and Loc14. These are trains in locations 2, 13, and 14 with train in locations 13 and 14 moving in anticlockwise direction while train in location 2 in clockwise direction.

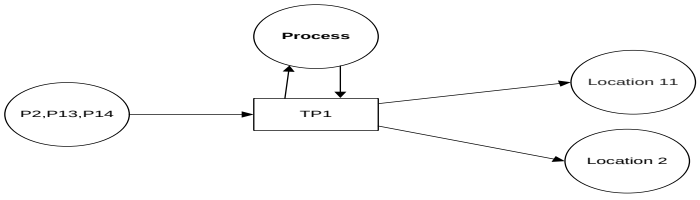


Figure 6-4 Delay and Schedule Model

This model in figure 6.4, the time and train schedules model will be analysed by checking some of the properties. The time will be analysed and the location or the destinations. Train in location 2 is meant to have priority over others in order to avoid deadlock. In this model, the bird is meant to be in the place named process. TP1 is a location of the sensor while the locations have a location sensor. Our locomotive or train is equipped with mobile or wireless device that can transmit the location, therefore location in this place represents wireless sensor. The location with all the trains can represent three locations as seen in figure 6.2 or a station where three trains that are moving in different directions (clockwise and anticlockwise directions) assembled. The scenario in figure 6.2 showed a situation where two lanes can join to one. The lane from location 9 and location 11 meets at transition TP1.

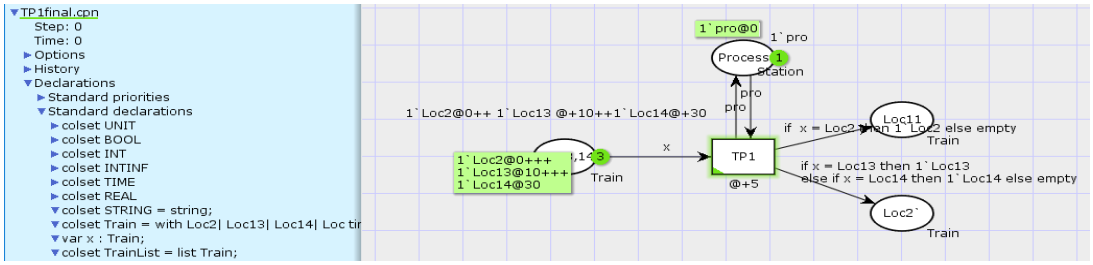


Figure 6-4a TP1 Delay and Schedule

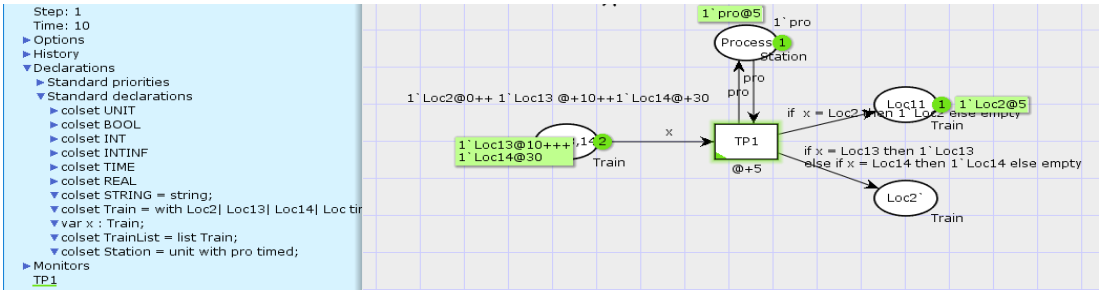


Figure 6-4b TP1 Delay and Schedule

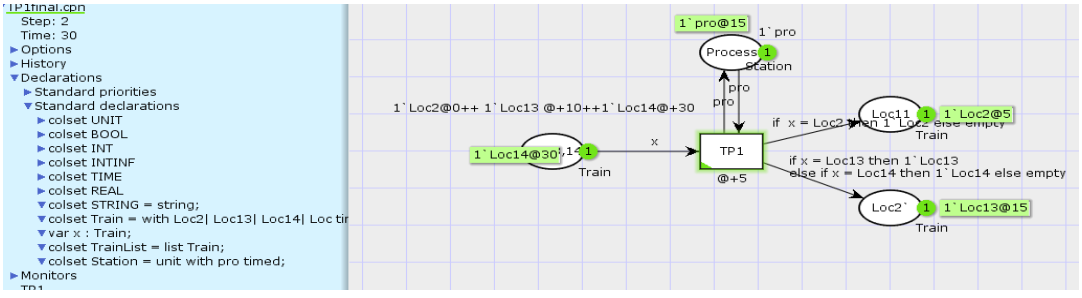


Figure 6-4c TP1 Delay and Schedule

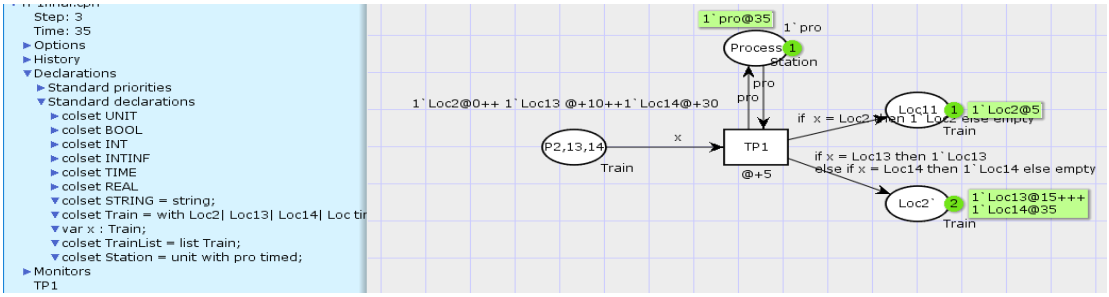


Figure 6-4d TP1 Delay and Schedule

Figure 6.4a showed three trains in location labelled P2,13,14. All the three trains are heading towards TP1 at the same time. This is a single lane track and the only way to avoid this kind of anomaly is to apply scheduling in the system. This scheduling will only allow one train at a time. In figure 6.4a, this was implemented in CPN with a place called process. The place or location called process is the location of a bird that will be monitoring the location sensor. The process location has 0 timed and one token in order to enable the transition TP1. The three tokens in location P2,13,14 were timed 0, 10 and 30. This shows that the train will only be available at this time. However, the location called process will see that only one train pass the lane at that particular time. The transition TP1 was also time for five allowing the movement to get to some point before allowing the next train within this time interval.

Figure 6.4b showed the first train as it was simulated at time 0 (zero) as seen in figure 6.4a. After the first train, the next train will only be allowed at time 10 unit as seen in figure 6.4b. The information on the left side of figure 6.4b showed the next train schedule time as 10. The first train moved in a direction to location 11 while the second train in the direction of location 2. Figure 6.4c was the third train which can only be allowed to move at the time unit of 30. This can be seen on the left side of figure 6.4c. The number 30 there is the time when the third train will be available. The direction of the first train is the same as the direction of the second train. The time interval was to allow them a good distance between each other. Figure 6.4d showed a total time of 35 as a result of time 5 interval that was added to the transition TP1.
In order to properly organise the trains from different locations access to a single track, there must be a queuing system. The queuing system will allow trains to wait for their time before proceeding to the next station. The queuing will be monitored by birds as well as monitoring the sensors. The queuing will be modelled in petri nets and will be simulated for results.

6.2.1.2 The Queuing Systems Model

If a train is in location 13, 14, 11 in anticlockwise direction and in location 2 in clockwise direction. This is a possible anomaly in the system that need to be detected early in order to avoid collision. In order to do this, the birds that monitor the sensors will put them in the queue. Birds communicate with one another in the group and across the groups. Hence, communication between birds will be active at all time. The communication algorithm will enhance the detection of such anomalies in the system. When a bird moves to a new location during "Move" call, it will locate the neighbours and groups. When the groups have been identified, every movement of the train in those locations will be identified and communicated to other birds. Therefore, queuing system will be activated as a security measure in this situation. The train at location 11 should be detected when in location 4 and redirected to follow TP3 because there is a train in location 2, 5 or 14 in a clockwise direction. If this condition is met (there is a train in location 4 in anticlockwise direction and a train in location 2 or 5 or 14 in clockwise direction), train in location 4 should be redirected to follow TP3 while priority will be giving to location ,2 or 5 or 14 in clockwise direction. This can be constructed in petri nets as follows;

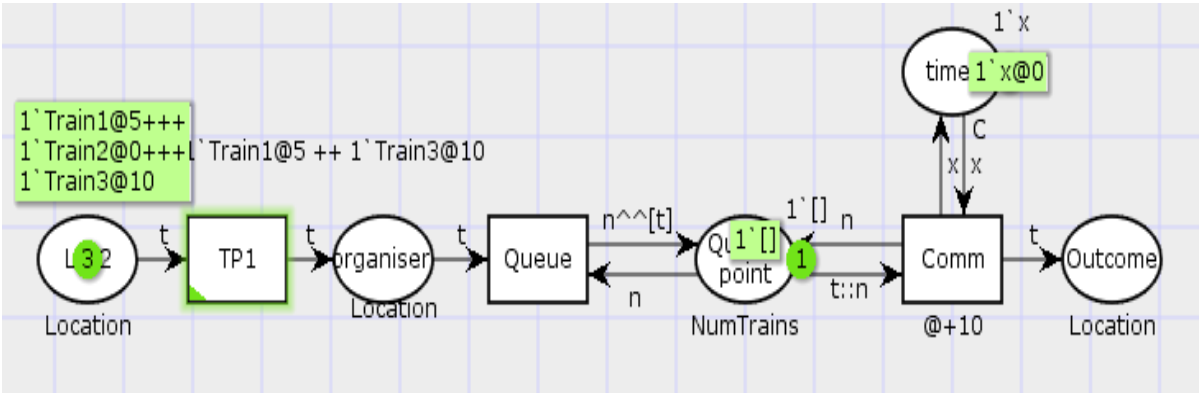


Figure 6-6-5a Queuing System Model

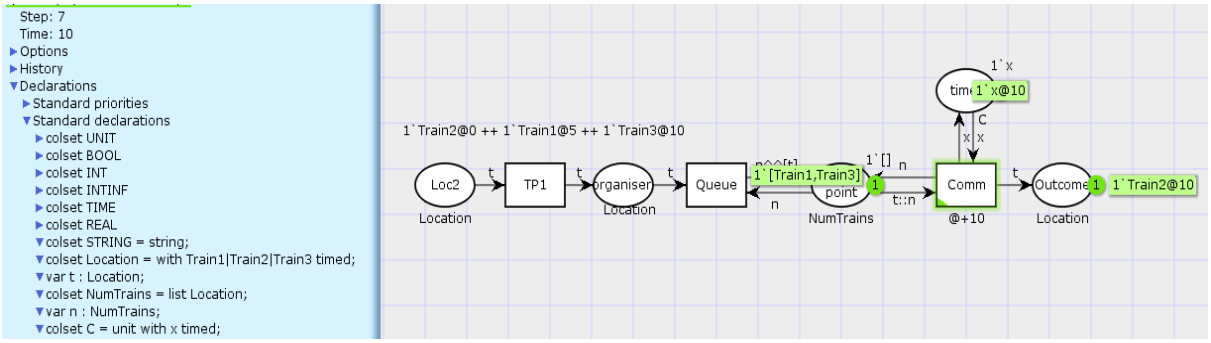


Figure 6-5b Queuing System Model Simulation

The tokens Train1, Train2 and Train3 represents the three trains explained above. These three trains are meant to meet somewhere in the track. However, the birds through their communication will detect it and activate the queuing system as seen in figure 6.5. The most important thing here is the detection of the anomaly. The reason behind modelling this queuing system is for the understanding of the anomalies and the dangers they create in the system. Once such an anomaly is detected in a system, the trains will be timed as seen in figure 6.5. Train2 was timed 0 which gave it priority over other trains. Train1was timed 5 and Train3 was timed 10. Train one as seen in figure 6.5b will move first while Train 1 and 3 will be in the queue waiting for their time as seen in figure 6.5b. This is to allow Train 2 access first in order to avoid collision or deadlock in the system. The place with the name Loc2 is the station where they converge while the transition TP1 is still the same. The place organiser will organise for the queuing system which will activate the transition queue. When the transition queue is activated, it allows the first train to pass which is Train2 by activating the transition comm. Train1 and Train3 will queue for their time in the place queuing point. Because of this, the place has an empty list []. The variable n represents the number of trains, which has the type list and will accommodate list of trains. The arc inscription n^^[t] represents the concatenation operator with n as a list and t coming in. The t will be added to the tail end of the list and the next is the t::n. The t is added to the list and the n is returned. This showed that only one train is added to the queue once at a time.

Since all the possible combinations of the collision in the first point of the Train scenario has been accessed. This assessment has revealed some risk points as well as what to feed to the bird as well as areas to monitor and the things to lookout for as anomalies. This first part has given an insight into the things to avoid using three trains.

6.2.2 TP2 for Transition 2

TP2 has places 2, 5 and 14 that are connected to it and since trains can move in both directions as indicated by the arrow signs. It means that trains in all the places can either move in clockwise or anticlockwise direction. The danger or anomaly that can be seen here is when a train in location 2 is moving in an anticlockwise direction while train in location 5 and 14 are moving in a clockwise direction. There is a possibility of deadlock or collision at some point. This possess a risk on the system and should be minimised. This condition should be avoided by a bird monitoring train in TP5 or before it. Before a train in a clockwise direction gets to transition TP5, a check will be carried out to know whether a train is in location 2 and 14. If a

train is in location 2 and 14, location 5 should be enabled to allow diversion and 2 should be allowed to proceed to the next location. After 2, location 14 or 5 should be allowed access with timed delay for each of them. This will allow the 5 or 14 to move first before the other one. This can be modelled as follow;

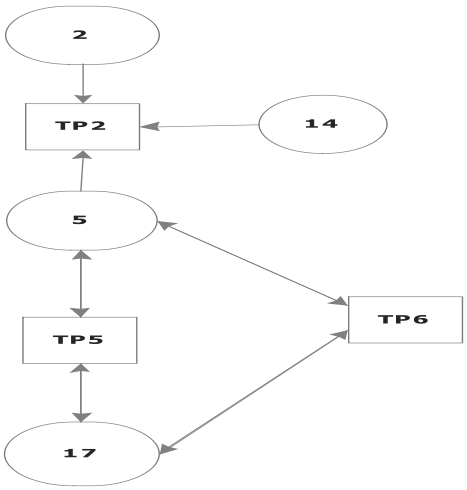


Figure 6-6 TP2 possible Risks Locations with TP2 and Solutions

There is a possibility of a deadlock in the system when trains are in locations 2, 5 and 14. Hence, in order to detect such deadlock before it happens, location 17 should be monitored whenever a train is there. If a train is at location 17, other locations (2 and 14) should be check for the presence of train. If there are trains in locations 2, 14, the train at location 17 should be diverted to TP6 to allow location 2 to go first before 14 and then 17. The train at location 17 will go around the track through TP6 and come back to location 5. The arrow that goes from locations 17 to TP6, TP6 to location 5, location 17 to TP5, and TP5 to location 5 are all bidirectional arrows. They showed that train can travel both ways. It is either from or to the location or TPs. A train in location 5 can go back if the need be, as well as train in location 17. Therefore, placing a bird in these locations can detect the combination of the above-mentioned anomalies and avoid it happening.

6.2.3 TP3 and TP6 for Transition 3 and 6

TP3 and TP6 are straight forward compared to others in more complicated conditions. The first thing to do in order to avoid collision is not to allow train movement in opposite direction. If this eventually happened that a train is diverted to TP6 as seen in figure 6.7, location 6 and 3 should be checked for trains. If there is a train in location 6 moving in anticlockwise direction, the train that is being diverted from TP5 to TP6 should be delayed in between TP5 and TP6 to allow the one in location 6 access first. However, a train can be in location 3 and heading clockwise while in location 6 is heading anticlockwise. The two directions should activate or

enable TP6 and delayed location 3. When TP6 is enabled, train in location 6 will go in through TP5 to allow the train in location 3 to proceed to the next location. The train in location 3 will go through TP5 and comes out through TP3. Although there should be another check when in TP4 heading to TP3 for location 6. If location 6 is free, TP4 will be enabled and access will be granted to the train heading to TP3 and location 6.

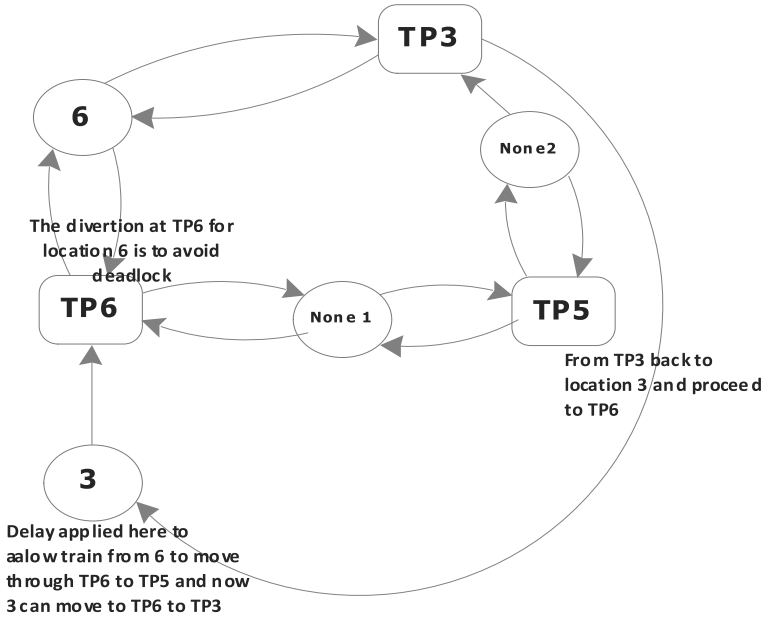


Figure 6-7 TP3 and TP6 Possible Risks Locations Model and Their Solution

6.2.4 TP4 and TP5 for Transition 4 and 5

Transition TP4 is connected to locations 11 and 4 as well a location in between TP4 and TP3 which will act as a buffer. Transition TP5 is connected to locations 5 and 17 as well as location between TP5 and TP6 which will act as a buffer. The only condition that can result in an anomaly in the system for TP4 is when two trains are moving in clockwise and anticlockwise direction between the two locations. A train can be in location 11 and heading towards TP4 in a clockwise direction while another train can be in location 4 and heading towards TP4 in an anticlockwise direction. There is an anomaly here because that shouldn't have happened. Two trains are not supposed to move in opposite direction in the same track, because it is an anomaly in the system. However, when such situation arises as in the case here, there should be diversion and delays to allow one access before the other. Hence, location 4 should be diverted to the buffer between TP4 and TP3. Also train from location 17 should be diverted to the buffer between TP5 and TP6. Since the trains can move backward and forward, they can stay in the

buffer until access to move to the next location is granted them. That means, it can either turn back to location 4 or location 11 and proceed with their journey.

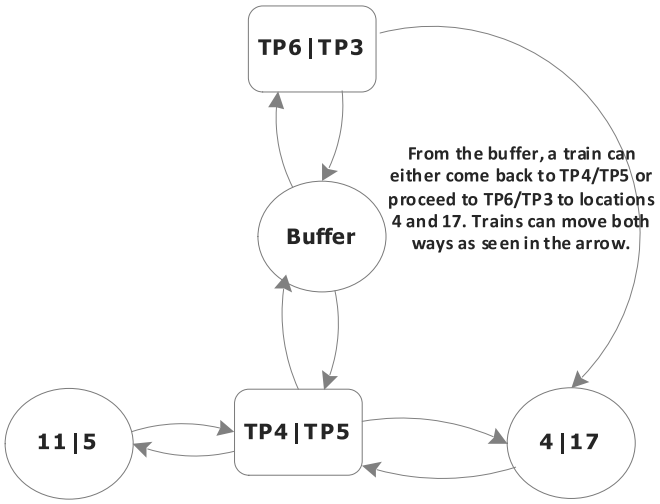


Figure 6-8 TP4 and TP5 Risk Location Model and their Solutions

6.2.5 TP7, TP8 TP9 and TP10 for Transitions 7, 8, 9 and 10

The transitions TP7 and TP8 have the same conditions that were explained for TP1 and TP2. TP7 and TP8 are exactly the mirror of TP1 and TP2. Hence, the conditions are the same as they represent the same thing. Trying to model the conditions in them will be a repetition of the same condition. However, focus will be placed on TP9 and TP10.

TP9 and TP10 can be modelled together as there is a similarity in the locations. Location 9 is the mirror of location 10 on the other side of the track. These two locations have the tendency of a train coming from locations 9 or 10 and heading to TP9 or 10. The train might be going to park at locations 13 or 15. If this happens to be the only train, it will go straight to the parking space. However, there might be a condition where a train is in location 14 or 16 and heading towards TP9 or TP10. In this case, there is an anomaly in the system that should have been avoided before now. Thus, when this happens, the arrows showed that train can either go backward or proceed forward. The train at these locations should be delayed and allows the train at locations 9 or 10 to proceed to their parking space. Hence, such an anomaly should be detected at location 2 and 7. When a train is in location 2 and 7, a check should be carried out by bird for the presence of train in locations 14 and 16 as well as locations 13 and 15 depending on the direction of movement. If 14 or 16 are in clockwise direction while 2 or 7 are in

anticlockwise direction, 14 or 16 should be delayed while 2 or 7 should be allowed to proceed to the parking space.

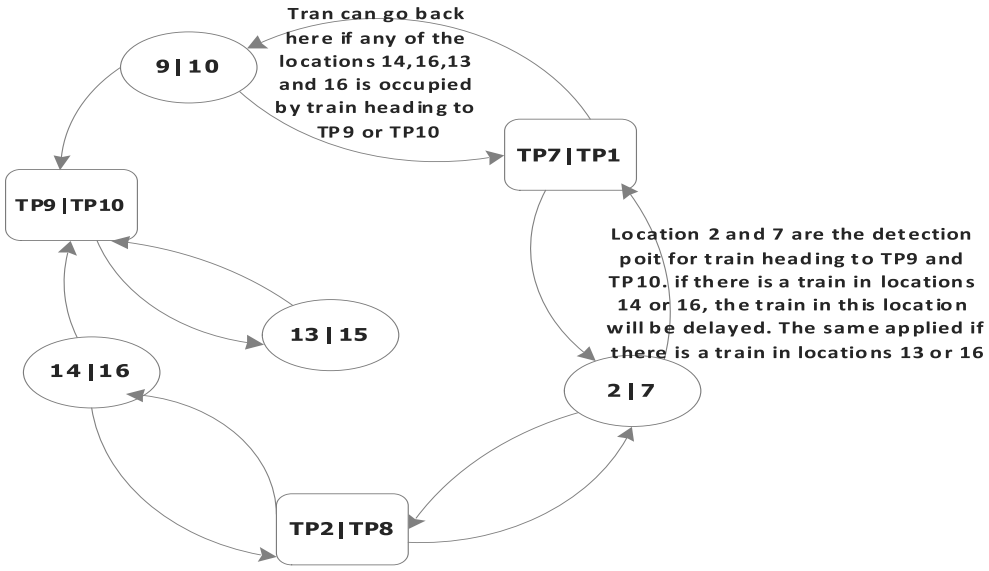


Figure 6-9 TP9 and TP10 Risk Locations Model and their Solutions

6.3 The Train Tracks System with Birds for Detection

Figure 6.4 to figure 6.9 are the mapping out of some important locations and scenarios. These locations and scenarios determine whether deadlock or accident will happen in the system. locating these important points in the system is very vital because, knowing the risks will help in devising a means of solving it with the algorithm. Therefore, risks locations and scenarios have been located and solving the problems in our train system using the bird detection model in chapter 5. From figure 6.4 to 6.9, devices are organised as groups. Figure 6.4 was modelled in petri nets as seen in figure 6.4a-d. This was simulated for results and to know how the system will run in a real environment. However, the result show that it is possible to place trains in such condition. The rest of the images shows the construction of the groups that was defined in the previous chapters. These groups will be monitored by birds and as earlier explained in chapter 5, every device in the system is regarded as a bird.

Since every device in the system is a bird; the sensors, the HMIs, PLCs and among others. However, the sensors are connected through the PLCs and the commands that goes to them are received from the HMI as see in chapter 4. Chapter 4 was the low level of the systems devices and their communications. It is the system design which has all the elements that is needed for modelling the system. Chapter 5 showed that every sensor in the system can be identified by

their names, address and they have status. The status will show whether they are ON or OFF. With the name and the address, a sensor will be located, and their status will be known and compared with the neighbouring devices in the system. The comparisons are the works of the bird as stated in chapter 5. These are the information made available to the birds that come in to the group. The state, status and the address of the sensors and birds that are in the group. Below in figure 6.10 to 6.14 are the train tracks with all the devices connected to the bird for monitoring.

6.3.1 TP1 and Group with Birds

Figure 6.10 is a group of devices which are connected to B1 and B2. These birds check for locations 2, 13 and 14 for priorities. These priorities were simulated in figures 6.4a-d. Without this, there will be deadlock in the system. This can cause collision in the system, but with the birds communicating with each other, B1 knows whether there is a train in location 2, 13 and 14. With this information, priority will be given to one of the train and delay also can be introduced as simulated in figures 6.4a-d. B2 is connected to the transition, which will tell the state of the transition. The state can either be open or close, true or false. However, both birds communicate, and decision is made as seen in chapter 5. It is evident that $B1 \in B2 \; and \; B2 \in B1$ which covers the whole locations and places.

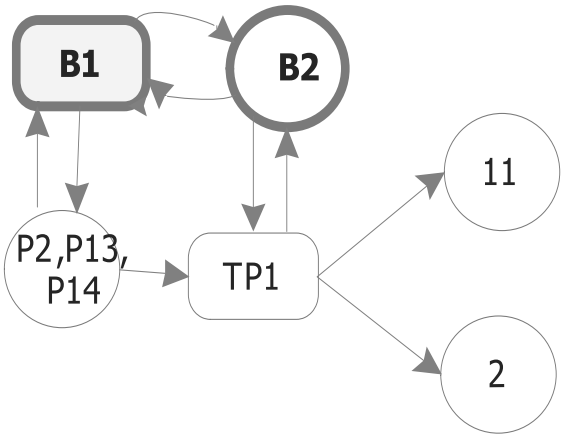


Figure 6-10 TP1 with Birds B1 and B2

6.3.2 TP2 and Group with Birds

The group in figure 6.11 has mostly bidirectional movement as shown in the arrows. The connections are the relationships between places and transitions. In figure 6.11, there are two birds with the ID staring with B in the system, birds B3 and B4 that connects other birds to transitions and places together. To stop a situation that was shown here in figure 6.11, B3 that was connected to all places talks to B4 that was connected to all the transitions. When a train

is located at location 17, B3 will check places 2 and 14 for a train. Location 2 is anticlockwise while other locations can be either clockwise or anticlockwise. However, in this condition, our train was run in an anticlockwise for location 2 and the collision happened just above location 5. The three trains are running at the same speed. Now the birds are monitoring the condition and if there is a bird in location 17, 2 and 14. The bird B4 will communicate with bird B3 and this will be flagged as an anomaly in the system with a pointer to these locations. Since B4 is monitoring the transitions, with such condition, TP5 will be enabled to allow access and B3 will apply delay to 14. The same is applied for any location here. There is a constant communication between B3, B4,2, 5, 14, 17, TP2, TP5 and TP6. All these forms a single group that are being monitored by agents (birds) as seen in figure 6.11. $B3 \in B4 \; and \; B4 \in B3$ which covers the whole places and transitions.

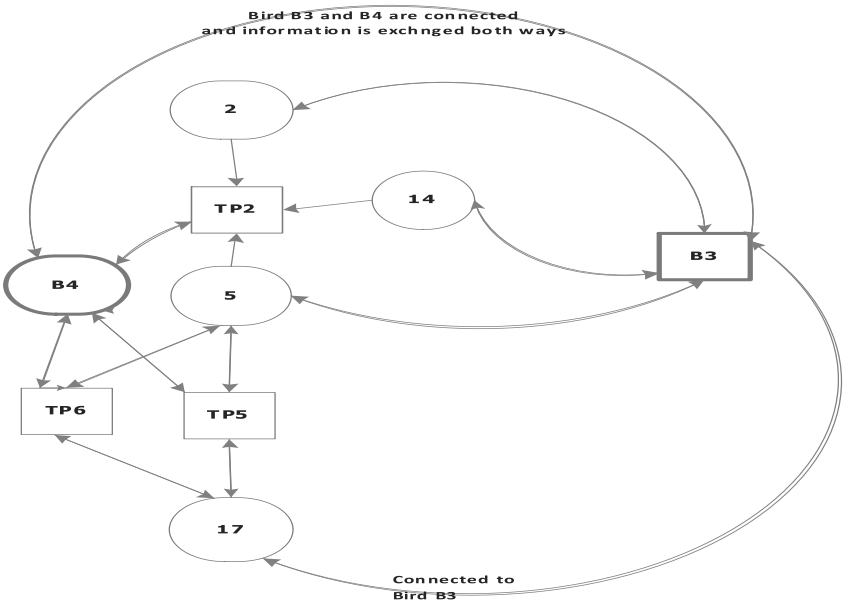


Figure 6-11 TP2 with Birds B3 and B4

6.3.3 TP3 and TP6 Group with Birds

The two important locations here in figure 6.12 are locations 6 and 3. The two locations are connected to TP3 and TP6. If a train is in location 6 and heading toward TP6 in anticlockwise direction, while another train in location 3 in clockwise direction, there is a possibility of collision. This anomaly was discovered in section 6.2.3 during the analysis. In the laboratory experiments, the two trains at the same speed met at the location immediately after TP6 in anticlockwise direction. This process was repeated for several times and the collision point is the same for all the testing for the speed. However, different speed shows different meeting points. This will be seen in another section. For location 6 and 3 in opposite direction, the only remedy here is to divert the train from location 6 through to TP6 and comes out to TP3 and

continues the journey if the place is free. Alternatively, the train can stay in the buffer until the one from three passed as seen in figure 6.12. These are constantly monitored by B5 and B6 that connects all the devices in the group. B5 checks the transitions to know the state of it and the B6 checks the locations and communicate it with the B5. If the criteria for diversion is met such as two trains in locations 3 and 6 in opposite directions. $B5 \in B6 \; and \; B6 \in B5$ which shows their communication and connection to each other

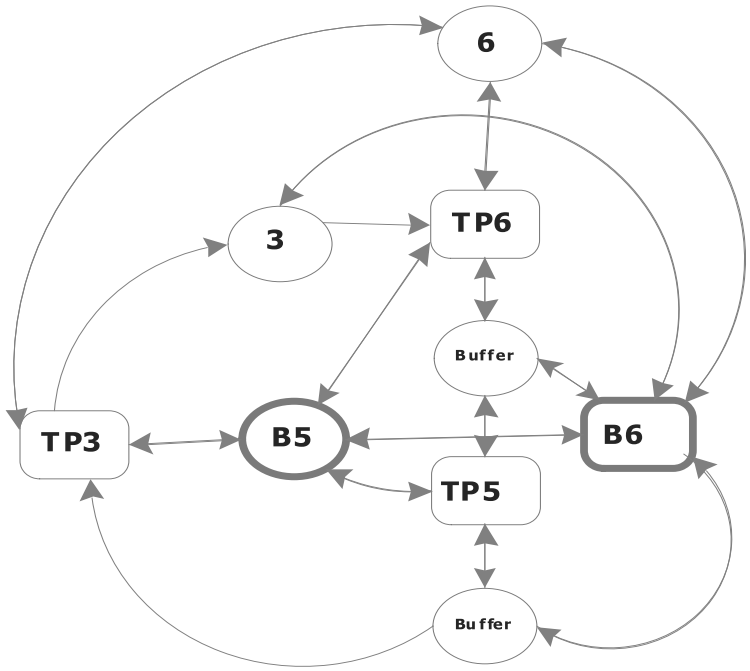


Figure 6-12 TP3 and TP6 with Bird B5 and B6

6.3.4 TP4 and TP5 Groups with Birds

These are actually two groups consisting of TP4 and TP5 as well as another group with TP6 and TP3. This shows the connection between these groups and the previous group in section 6.3.3. with the information flowing from all the groups, there will be communication between the groups in the system. Simulating the group with TP4 and TP5 as seen in figure 6.13 with trains from opposite direction. The outcome of the simulation is that trains meet at the same point under the same condition such as speed and direction. The processes are being monitored by B7 and B8. B7 monitors the transitions while the B8 monitors the locations. A train in location 17 in clockwise direction and another train in location 4 in anticlockwise direction will tiger the B7 to enable TP4 so that the train in location 4 will wait in the buffer to allow the train

in location 17 to go first. Since the train can go in forward and backward directions, the train can come back to TP4 to location 4 and proceed to location 11. However, it can also continue from TP3 depending on the destination of the train. The same way if trains are in locations 11 and 5 in opposite direction, where 11 is moving in clockwise direction while 5 is moving in anticlockwise direction. When these two scenarios are detected, the 11 has to be delayed, allowing the 5 to move to the buffer between TP3 and TP4. Although this can take time, but the two have the same length and there must be priority. $B7 \in B8 = B8 \in B7$ meaning there is a communication between them in the system.

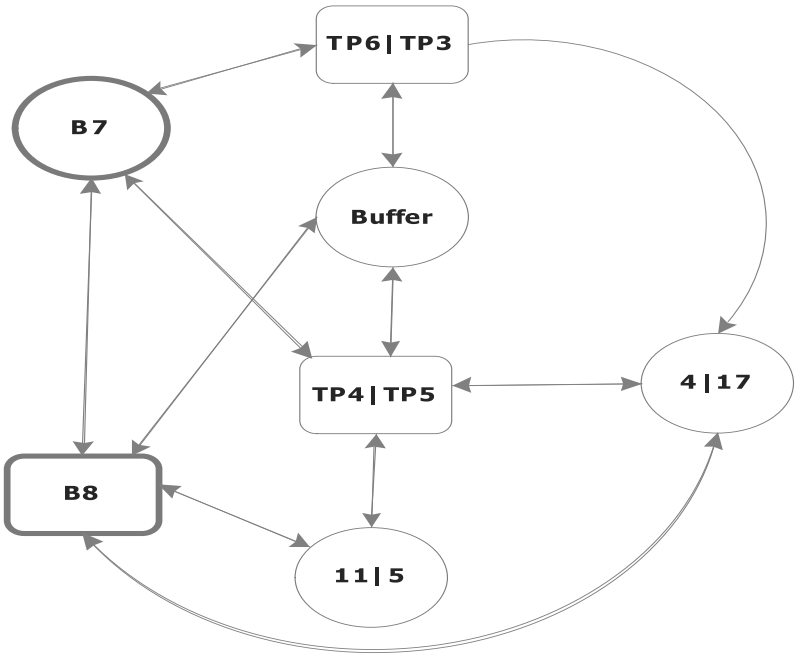


Figure 6-13 TP4 and TP5 with Birds B7 and B8

6.3.5 TP9 and TP10 Groups with Birds

The group in figure 6.14 is one of the largest groups with TP9 and TP10 as the primary transitions. The group is connected to TP1 and TP7 as well as TP2 and TP8. The first group comprises of TP1 and the second group is made up of TP2. This is what is defined in chapter 5 as the birds from one group belonging also to another group. This group is connected with B9 and B10 throughout the group. B9 is connected to all the locations while B10 is connected to all the transitions as well as B9. The possibilities of an anomalies in the system was played using two trains. One train was placed in location 2 heading to TP9 while the other train was placed in location 9 heading to TP9. This should never have happened, but in this instance, it is an anomaly in the system. Therefore, B9 will detect the presence of train in location 2 and location 9 in opposite direction. Hence, B10 will enable TP9 for train in location 9 to enter and wait for train in location 2 to go first. This scenario is similar in TP10 when a train is in location

7 and 10 in opposite directions. B9 will detect the presence of train in these two locations and in opposite directions and enable the TP10 for location 10 to go in and wait for 7 to pass. When these conditions are met, it will enable the opening of either TP9 or TP10 to avoid collision in the system. These scenarios were played several times for TP9 when it was close to between TP9 and location 14. The collision for the transition TP10 was also between TP10 and location 16. $B9 \in B10$ *and* $B10 \in B9$ which shows a healthy communication between the groups and the monitoring locations and transitions in other way $B9 \in B10 = B10 \in B9$.

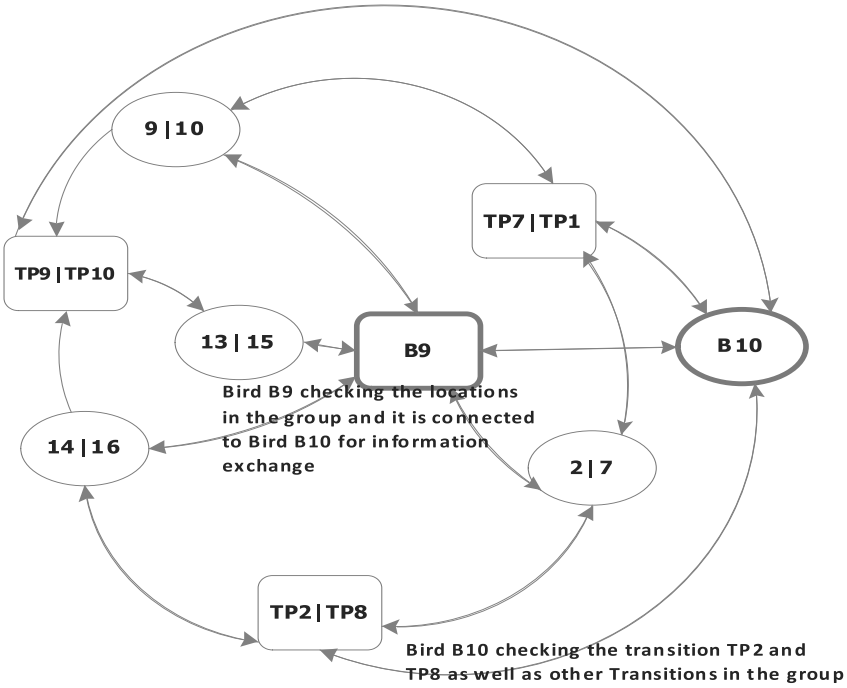


Figure 6-14 TP9 and TP10 with Birds B9 and B10

6.4 Bring it All Together for Train Track System and the Birds

Figure 6.15 shows the communications among the groups and how they are connected to each other in the system. It is evident from the direction of the arrows that no group is left on its own in the system. It shows the interconnections between the groups. Some devices belong to more than one group as indicated by the arrows. B1 in the first diagram communicates with TP1 and B2. In the same way, TP1 belongs to the last group which is figure 6.14. B1 in the first group as seen in figure 6.10 communicates with locations 2, 13 and 14, location 2 belong to group in figure 6.11 as well as group in figure 6.14 with locations 13 and 14. Location 11 in figure 6.10 is also in figure 6.13. The transitions TP5 and TP6 in figure 6.12 are also in figure 6.13 which communicates with B5, B6, B7 and B8.

This approach in communication is what made the flock of bird detection stronger compared to other species of animal. Their aerodynamic movements in the sky are determined by their communication as stated in chapter 2 and 3. The diagram in figure 6.15 has shown that each group in the system interact with one another in one way or the other according to this analysis.

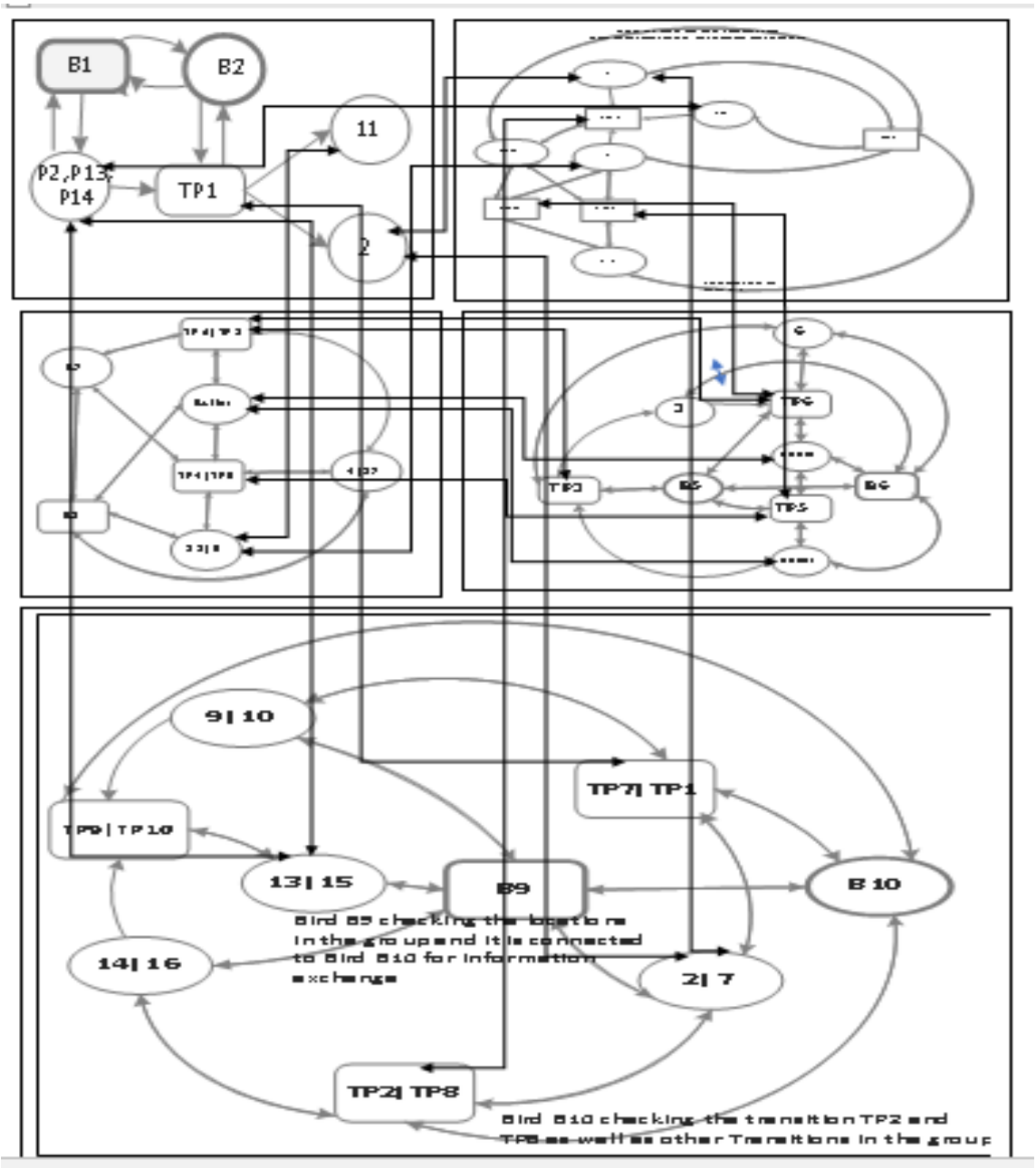


Figure 6-6-15 Communication among the Groups

6.5 Analysis of the Collision Points in the Track

Below is the analysis of the time it takes for the trains in the same track to meet as well as their meeting point in the track. All the groups above will be plotted against time and speed of the

train as well as their meeting point on the track. The more power is applied to the track, the more the speed of the train will increase. Based on the fact that the model used in this experiment is from Hornby and the controller is R8250 which serves the purpose at this time, the speed is controlled by increasing the power from the controller. Here the speed is kept at constant by maintaining the same level of power on the track at 12. The trains were travelling at a constant speed of 1.2 kilometre per hour. The time for each train at the designated locations will be recorded.

60 minutes ----------> 1.2 km

1 minutes ----------> 0.02km = 20 metre

6.5.1 Group 1

Table 6.1 and group1 graph are the representation of the analysis and experiments that was carried out to determine the collision time as well as the point on the track. Train moved from location 2 to 11 in different times as seen in table 6.1. This process was carried out for several times and the best times were chosen for five times. The same with locations 13 and 14 from TP9 to TP1. The average time for location 2 is 2.02, location 13 is 1.78 and location 14 is 1.81. The average time it takes from location 2 to TP1 is 1.73 and from there to location 11 was around 0. 29 and these times are in seconds. The graph for this group 1 identifies the meeting point in time between 1.78 – 1.80. This will be a point after TP1 in clockwise direction which is between the distance of 1 and 2 but closer to 2. Hence when a train is coming from location 13 or 14 and heading to location 2, the time it will take it from TP9 to TP1 is known. The train can be delayed, allowing the one from location 2 to move first before the one from location 9. Collision point can be predicted with the data in table 6.1 and the graph for group 1.

	Location2	Location13	Location14
1	1.98	1.73	1.88
2	1.93	1.78	1.82
3	2.15	1.78	1.76
4	2.07	1.76	1.78
5	1.98	1.81	1.82

Table 6-1 Group 1 Trains Distance and Travel time

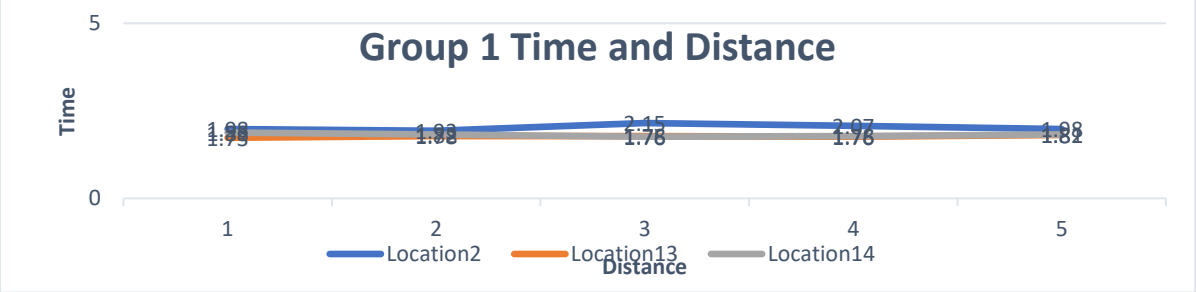


Figure 6-16 Group 1

6.5.2 Group 2

There are three important locations in this group; location 2, location 5 and location 14. The average travel time of train from location 2 to location 5 is 2.55 seconds. In the same way, a train from TP5 to location 2 has an average travel time of 3.87 and from location 14 to location 2 is 3.89 based on the information in table 6.2. This showed that the points of collision in this situation is after TP2 in anticlockwise direction. That means the train from location 2 will collide with train from location 17 when in TP5 after location 5. The train from location 2 is in anticlockwise direction while location 5 is in clockwise direction. Hence, the diversion should occur before getting to location 5 for a train from location 17 heading towards TP5. Graph for group 2 shows the location at which this can happen. The distance for location 5 and 14 are longer compared to location 2. Collision will be more towards location 2 as the train there will not travel long before collision happens.

	Location 2	Location 5	Location 14
1	2.18	3.87	3.66
2	2.32	3.86	3.89
3	2.24	3.99	3.89
4	2.26	3.81	3.94
5	2.24	3.80	4.08

Table 6-2 Group 2 Trains Distance and Travel time

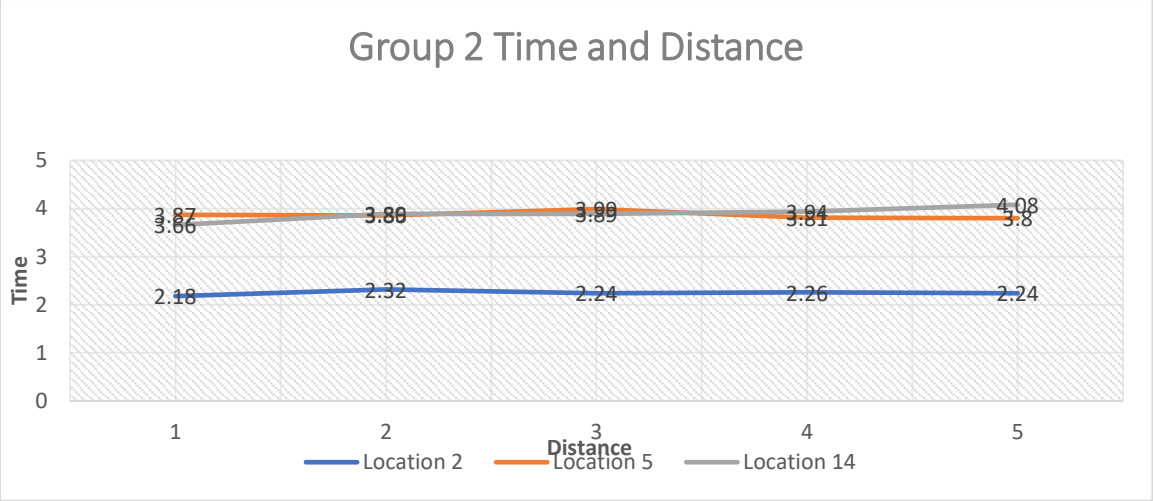


Figure 6-17 Group 2

6.5.3 Group 3

In group 3 are two locations that are very important which are locations 3 and 6. The train time will be measured from location 3 to location 6 in anticlockwise direction. The time will be measured also from location 6 to location 3 in clockwise direction. The best condition in these locations are never to allow two trains on the opposite sides of the track. However, it would be a good idea to know the travel time of the trains from one location to another in order to determine their collision point when such situation arises. Below in table 6.3 are the data that was collected in the lab during the analysis of this aspect of the train movement. It is obvious that there will be a collision when two trains on the same track are moving in both clockwise and anticlockwise directions. The collision point was before TP6 and TP3 as seen on the graph. The average time from location 6 to location 3 was 8.49 while location 6 was 8.28.

	Location 3	Location 6
1	8.65	8.33
2	8.61	8.06
3	8.25	8.17
4	8.58	8.61
5	8.38	8.21

Table 6-3 Group 3 Trains Distance and Travel time

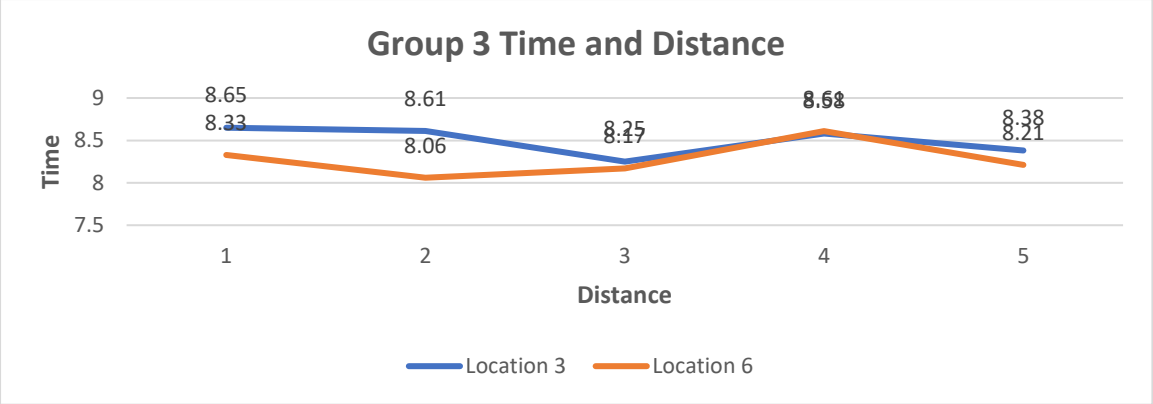


Figure 6-18 Group 3

6.5.4 Group 4

Group 4 are two groups together which each of the group is a mirror of the other. They represent the same condition and similar track lines on the other side of the track. Location 17 is the same as location 4 on the other side of the track. In the same way location 5 is the same as location 11 on the other side. Data was collected in order to justify the collision point and the travel time. Table 6.4 is the data collected and the graph below was plotted based on the data. However, only one table as well as graph will be used to represents both groups. A train from location 5 heading to TP5 and to location 17 as well as train from location 17 heading towards TP5 and to location 5. The collisions here happened after TP5 in clockwise direction, while between location 11 and 4, collision happened before TP4 in clockwise direction. The average time for this travel for location 5 was 2.39 while location 17 was 2.47.

	Location 5	Location 17
1	2.45	2.56
2	2.30	2.43
3	2.22	2.40
4	2.58	2.60
5	2.42	2.38

Table 6-4 Group 4 Trains Distance and Travel time

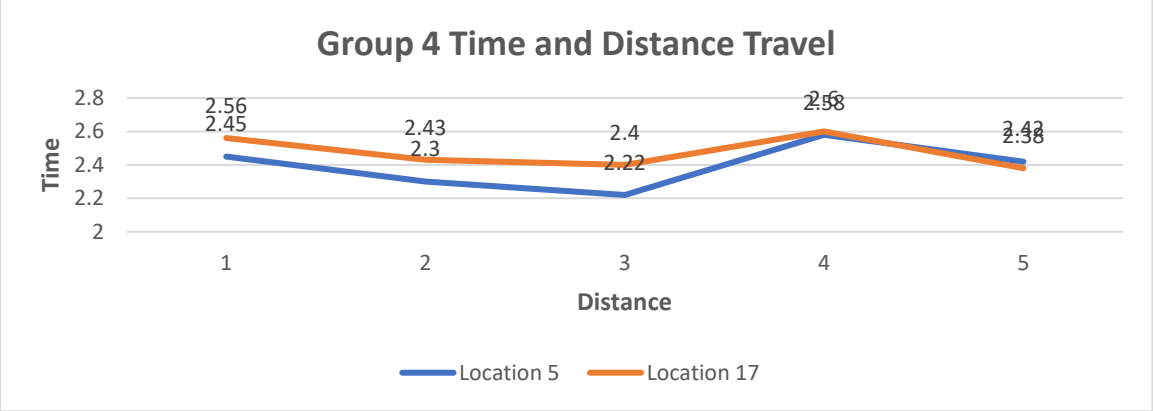


Figure 6-19 Group 4

6.5.5 Group 5

Data collected here are for trains travelling from location 2 to location 9 in a clockwise direction and from location 14 to location 9 in anticlockwise direction. The average travel time for train from location 2 to 9 was around 3.44 while from location 14 to 9 was 2.47. Since the distance from location 2 to location 9 is longer than that of location 14 to 9, the meeting point was after the location 9. The train from location 14 will meet the other train after it has crossed the location 14. This will happen at about 2.96 seconds which shows that the train from location 14 must have crossed the location 9.

	Location 9	Location 14
1	3.46	2.34
2	3.46	2.52
3	3.37	2.49
4	3.42	2.48
5	3.53	2.52

Table 6-5 Group 5 Trains Distance and Travel time

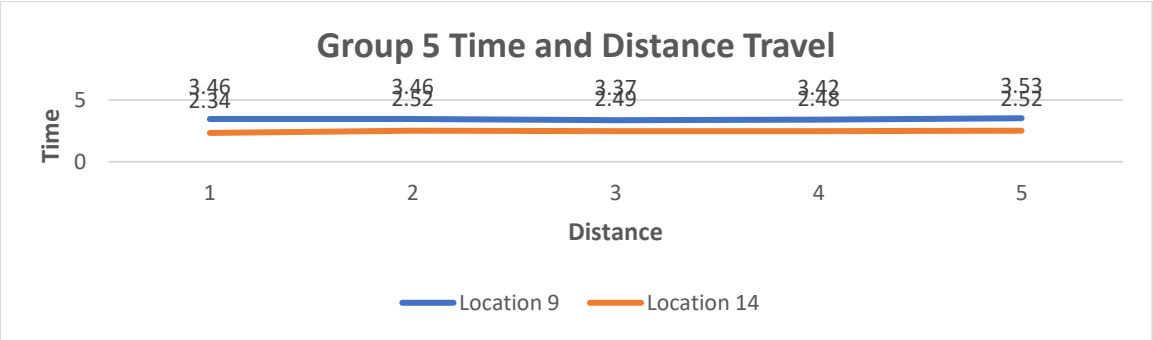


Figure 6-20 Group 5

Summary

This chapter has delved into the modelling and analysing of the case study that was introduced in chapter 3 and 4. This was a single case study in order to test the product of this project. This project produced a model that can be applied in a SCADA environment for the detection of anomalies in the system. These anomalies can range from human error to system malfunctioning as well as from external threats trying to penetrate the system and course problems. This was modelled on a train system that was used as a case study. The train system was equipped with all the devices of a SCADA system. The model was able to map out some risk points in the system. These are the locations that will trigger checking and monitoring of the points that are associated to it.

The train system was modelled in a way that the whole system was grouped. The grouping was part of the model that was introduced in chapter 5 and the main product of this work. Grouping of the devices made it easy to see clearly in terms of movement of the trains and their times as well as their locations. These locations and conditions were simulated in petri nets and the results showed the possibility of such conditions. Birds are dispatched into those groups and they communicate with the devices and with other birds in the system. The model show that a bird can belong to more than one group and a device can belong also to more than one group. These birds communicate with one another and exchange information for the system security. The model of the train with the birds showed that there is communication between groups and birds can move to different groups in the system.

Data was collected at the analysis of the train movement as well as time and location analyses were carried out. The analyses lasted for 8 hours which was spread over 4 days, each day 2 hours. The timer was used to capture the right time it takes the train from one location to another. The data collected was used to plot the graph and some are entered in the table. The analysis and data collected was to ascertain the collision point for this case study train system. The results showed that with time and distance, the collision point will be derived and can be avoided based on the speed of the train at each location.

CHAPTER VII

7 EVALUATION

The model that was produced in chapter 4 is the general model for the detection of anomalies on ICS. The approach there was using the protocols from the field devices for the detection of these anomalies. Protocols are divided into five pats and analysed by six birds based on six birds' approach. The class and sequence diagrams included Hadoop framework and cloud computing that constituted the so-called No-Trust-Zone (NTZ) architecture. The NTZ architecture was modelled in a way that every activity will be analysed by the bird in the cloud. Every event in the ICS such as open or close, will be analysed by the bird to determine whether it will cause the system to be unreliable or security issues. The idea was to show that the problem source can be internal as well as external. This form the general and generic approach of the bird's model. The second model was used in the case study where every device is regarded as bird.

The model in the case study in chapter 5 and 6 was the case where everything is regarded as bird. In this case, the HMI, PLC and sensors were all regarded as birds and their states were required, whether they are ON or OFF. These types of birds are static and do not move in their environment. However, there was another type of birds that moves in the environment. These are the type that do the detection and receives reward in their environment. They have groups and move around to different groups. The birds will detect the possibilities of event such as collision happening in a system. Communication between birds in the flock as seen in the literature is very fast that made them move in the aerodynamic form and still sticks together. This was one of the properties that was modelled into the system where a bird can belong to more than one group. This was called free coloration of information as seen in birds as they move away from predator.

This chapter deals with the evaluation of the models and to find out its efficiency and suitability for the purpose. The two models will be evaluated, the UML model as well as the model with the petri nets. However, the two models were to illustrate the robustness of the idea put together in the models. Hence the UML generic model of the bird's approach will be evaluated first followed by the petri nets model used in the case study.

7.1 Evaluation of the UML Model

A number of studies and literatures have recommended some approaches that are suitable for analysing UML model such as Moreno and Merson, (2008), that implemented a model driven engineering in PACC starter kit for model analysis and code generation from the model. This was for the performance prediction of models. Other systems such as Communication Extended Time Automata IF (CETA-IF) from Ober, Graf and Ober, (2004). Their approach was mainly generation and generation of model to IF and some description and introduction of some important properties. They showed the way to analyse them such as the introduction of observer and time properties. These produced a way of enhancing some of the analysis and verifications of the models. There are quite a number of literatures that deals on model checking as well as the analysis. Hence, one thing that was deduced from all the approaches is that there is no concreate way of doing it. Everyone implements and analyse their model based on what they think is best for them and for their system.

Below is the approach that would be adopt in analysing and evaluating the UML model in chapter four above. The model is made up of use case diagram as well as class and sequence diagrams. The data collection in chapter four depicted the important of time in data transfer between devices. This will be considered in the evaluation when drawing up the evaluation plan for the model. The evaluation of the UML model for the performance or the performance evaluation of the model will take into consideration the following.

1. The architecture will be evaluated for performance using a Queuing Network (QN) that supports concurrency.
2. The Use case diagram will be redesigned with annotations for references and better understanding of the flow of information.
3. The activity diagram for the system will be modelled in order to derive the right algorithm for the architecture (for both the general architecture and the bird)
4. Deployment diagram will be constructed for a better flow.

7.2 Use Case Diagram

The use case diagram in chapter 4 figure 4.29 was the initial use case that was use during the requirement gathering. The use case does not have the environment as an actor and the way the actors were organised made it difficult for the use cases to be read. The diagram in figure 4.29 does not have the Hadoop with a composite relationship to the cloud as seen in figure 7.1 with the arrow head to the cloud.

Figure 7.1 went further in explaining the detection by outlining the approach as the use case. The use case diagram in figure 4.29 has a use case labelled “detect activities”. These activities were listed in figure 7.1 based on packet that is coming into the system. The packets come from a Modbus device which for our system is from Schneider electric device or packet from siemens devices which is ProfiNet or S7 protocol. There are six use cases which formed the approach to the detection of anything on these two protocols. The protocols are passed on to the bird by the passer that was written earlier in chapter 4. The environment actor as well as cloud and Hadoop were given square boxes because they are interfaces. They do not belong to the bird algorithm, but they are vital in explaining and making the system understandable.

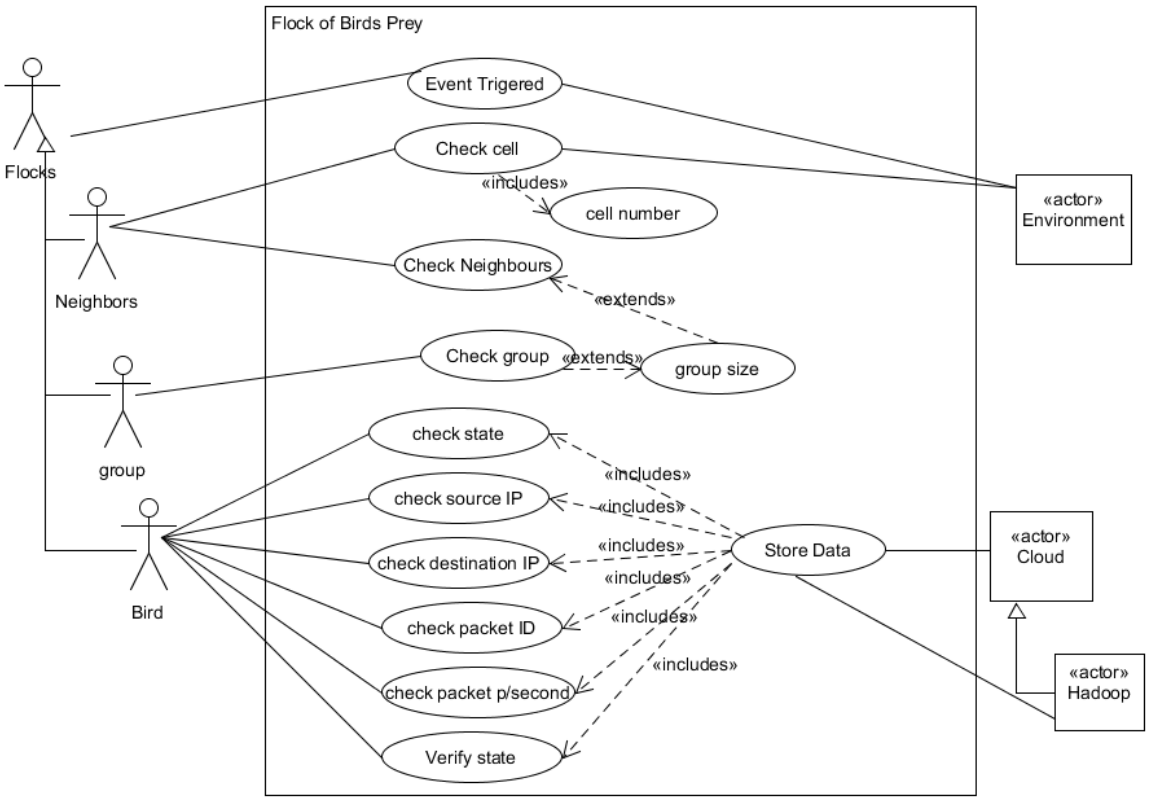


Figure 7-1 Modified Use Case Diagram

The UML diagram in figure 7.2 is the annotated diagram that explains in detail the workload of the system. This is an open workload as the protocols arrives from the external, as seen in the use case, the actor bird is triggered by the operation in the environment. The field devices and other sensors are part of the environment and any electrical pulse to the system will trigger the birds which is the algorithm. This has been annotated as an open system with initialization of the bird to 200. This can vary from system to system because systems capacities differ. When there is a trigger in the system, the neighbours are checked which forms the group that will sort out the packet that enters the system. Neighbours and groups are based on the cell.

Cells can contain maximum of six birds and there is a possibility that a cell might not have up to six birds in it. The criteria being that, in order to be able to analyse a packet, the number of birds in a given cell must be equal to six. This will be illustrated in the activity diagram as one of the activities of the algorithm. Checking the neighbours will be from east, west north and south based on the cell number. If this is complete, they will be regarded as group. This group of birds from a particular cell will now do the work that have been predefined based on the position in the cell or the cell number. The actor bird which is among the group will check the packets. The events are listed under the bird as the group will go parallel to analyse the packet. The actor's environment, cloud and Hadoop as annotated are all interface. The system on which the algorithm will be executed is regarded as the environment. The birds will be sent to all the sensors in a continual and consistent way. Every move of the system (birds) is to locate a sensor by a group in the cell. The actor environment has 50 packets arriving in the system per second based on the data collection in chapter 4. The delay in the annotation is based on the arrivals on the resources that are used such as the computer. The same goes to the actor cloud and Hadoop with events from the apache flume and HBase. Data from the sensors are channelled to the algorithm in the cloud by the flume and to the HBase and saved in the HDFS in Hadoop. These are events that adds time delay to the protocols. The travel time can determine whether the architecture is worth implement. If the latency is noticeable, it does not worth implementing because it might create another problem. Annotation in use case is a head start for performance evaluation.

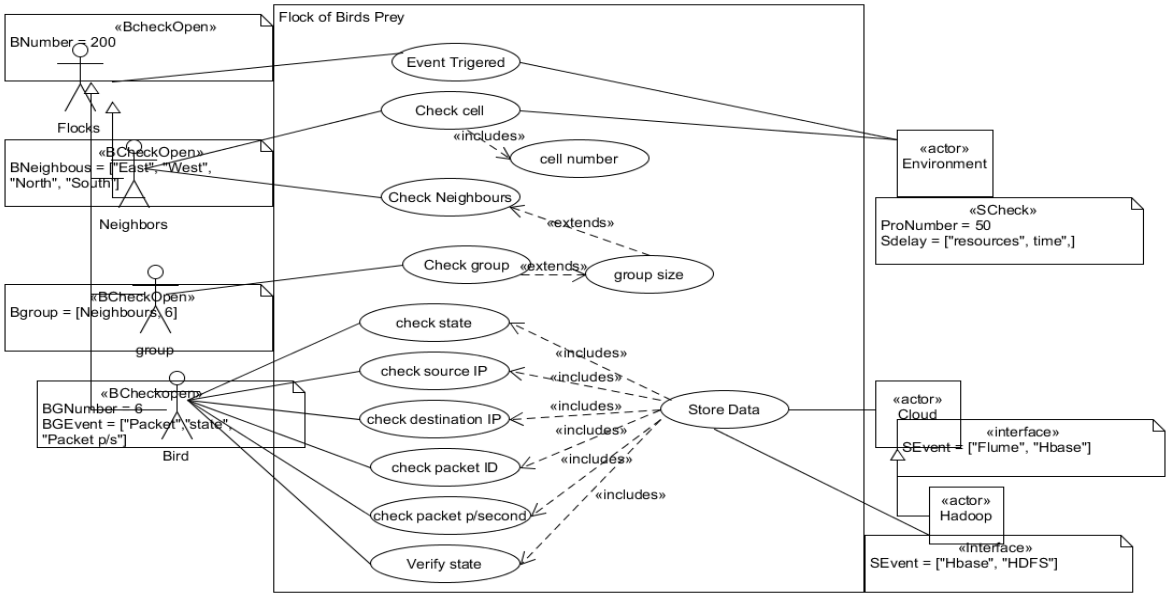


Figure 7-2 Annotated Use Case Diagram

7.2.1 Transforming UML Model to Queueing Network Model

The annotated use case in figure 7.2 was for the performance evaluation of the system. However, having an effective approach of evaluating a software system such as the one described above can be difficult. Queueing Network (QN) has been proving to be effective in performance analysis and evaluation (Lavenberg, 1983; Ebert, et al, 2017). The UML model has the actors which are translated to the QN model as seen in figure 7.3 (Marzolla, Mamprin, and Balsamo, 2004). Actors corresponds to request and response such as group of birds that search the packet. In the activity diagram of the UML model which shows the software interactions for the QN model. The deployment diagram of the UML model shows the interaction of the software with the environment such as hardware. The parameters seen in the QN model can be equated to the tags in the UML model. Hence, in order to carry out a performance analysis of the model, annotated activity diagram and deployment diagram will be created. Before this, a way should be created on interpreting the UML model to QN model as seen in figure 7.3.

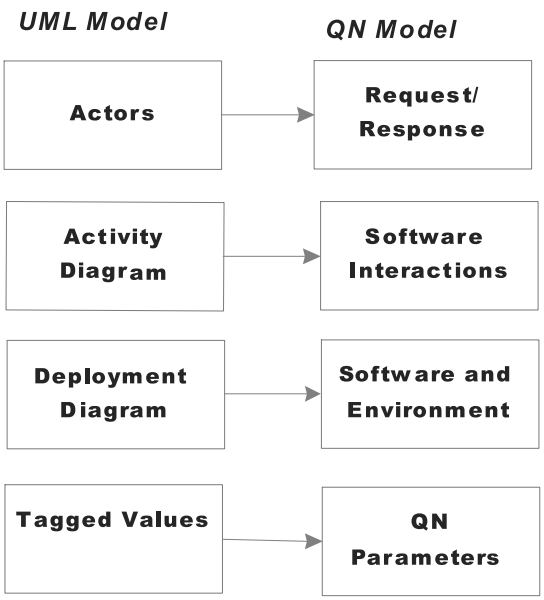


Figure 7-3 UML to QN Model

7.2.2 Annotated Activity Diagram of the Bird Algorithm

The activity diagram represents the interaction between software that was created or to be created. This shows the interaction between objects and methods in the software. These are the activities that are performed to achieve a purpose for a system. The activity diagrams in figure 7.4, 7.5 and 7.6 are the entirety of the system activities. The first activity diagram in figure 7.4 is the annotated activity diagram for the bird's software.

The annotation such as “Bstep” described the steps or the activity that will be performed. The step such as “Bdemand” is stated in the first activity as unbounded because it is not bounded to any time. The system is continuously running as far as the system is in service. The “Bhost” is the resources in which the activity is being carried out. Some of the activity are annotated under “Bdemand” as step and time. Step here represents the activity and time is the time of the activity. The time represents the time preprogramed in the system for the activity to happen. The activity ‘move’ function under heartbeat which is in milliseconds depending on the capacity of the system on which it will run. The “BmaxNo” represents the maximum number of birds a cell can contain while the Bhost = AL represents both the host and the artificial life. The artificial life prescribed as “AL” are the birds in the system. Some “Bdemand” are to count the number of birds that will form neighbours. This can be determined by defining a count and the direction to count from. However, it was made easier here by using cells and cell capacity as well. There are birds with limited TTL which is Time-To-Live and there are some with infinite TTL. The one with limited TTL die and the one without do not die. If action is activated for the birds and the cells were checked from left to right, the first that meets the requirements will be executed. They will go through the packet as programmed and check for any anomaly as well as verify the state of the system, whether the action will cause problem such as malfunction, unreliability and other operational issues to the system.

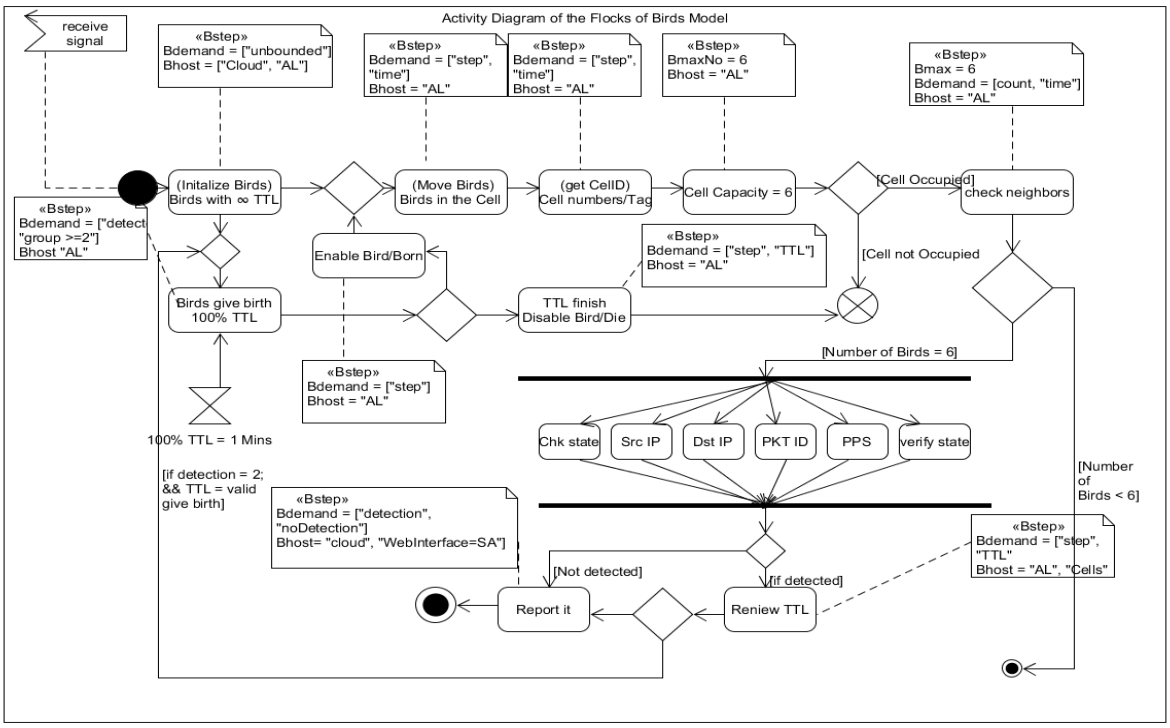


Figure 7-4 Activity Diagram of the Bird

7.2.3 Annotated Activity Diagram of the Network

Figure 7.5 is the activity diagram of the network. The fact that the algorithm will make use of the network for transfer of information, the activities must be documented and planned for. The network can add extra delay to the system or the software, therefore it is very important to recognise that in order to plan for it. This will give a clear picture of where the extra overload is coming or where an adjustment is needed in the system.

The activity starts by initializing the information with the field devices. This captures the activity between the software and the network. The bird algorithm will act on the information that came out of the field device. The information is sent in the network through the network switch, Tap and routers to the receiving end. The receiving end here is the cloud computing which is as well connected to the network. The bird algorithm is located in the cloud or directly connected to the network Tap from the Switch. When this information is received in the cloud or directly by the algorithm, it is acted upon. These are stated in the activity diagram in figure 7.5 as “Bstep”. The activities have different host stated as “Bhost” where the activity happens.

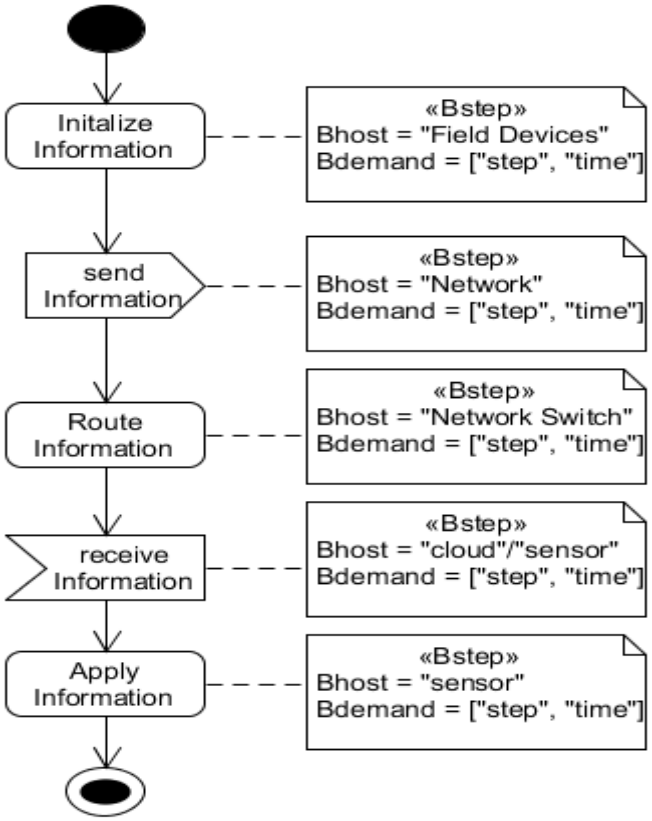


Figure 7-5 Activity Diagram of the Network

7.2.4 Activity Diagram of the Service Systems

Figure 7.6 is the annotated activity diagram of the service stations such as the server that hosts the cloud and Hadoop. The activity started with the data channel which is one of the activities that happen in the server. Data is channelled by the apache flume based on the configuration. The data is then received in the server based on the configuration. Our system was configured in a way that the HBase is in a standalone mode. Hence when the data arrived in the cloud, it arrives at the HBase memory. In the memory also is configured the apache Spark for a live stream of data. AL algorithm is integrated to Spark through its API and data is analysed on the process. After the data has been analysed, it is saved in the system. Through the data that was collected, data from the environment to the cloud has some delays in it. Through the annotation, it was easier to mark and recognise the steps it took as well as the parameters for determining the duration.

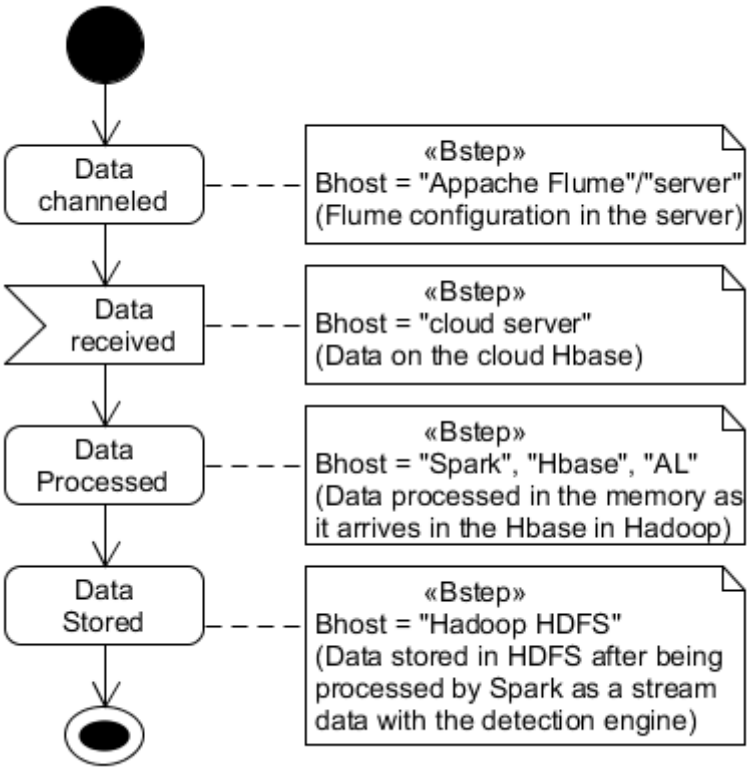


Figure 7-6 Activity Diagram of the Service Stations

7.2.5 Annotated Deployment Diagram of the System

The information that was collected during the data collection can now be used for determining the operational capacity and efficiency of the system. Deployment diagram in figure 7.7 is

annotated with some parameters. The 20ms is in millisecond and based on the data collected over time, the average data from the device to system is between 19.503 to 20.228 which was rounded up to 20 milliseconds. The policy that was implement in the algorithm is First Come First Serve (FCFS). One group can only take up one service or packet at a time. It takes the algorithm based on the assumption, 10ms to analyse a packet which causes the time increased to 30ms. The data before being analysed by the algorithm, passes through the switch to the algorithm and the time based on the information collected was still the same, 20ms. From the time the packet went through to the algorithm to the time it was saved, was in the approximation of 100ms rounded up. The deployment diagram with the annotations however, made it easier to determine the performance of the model.

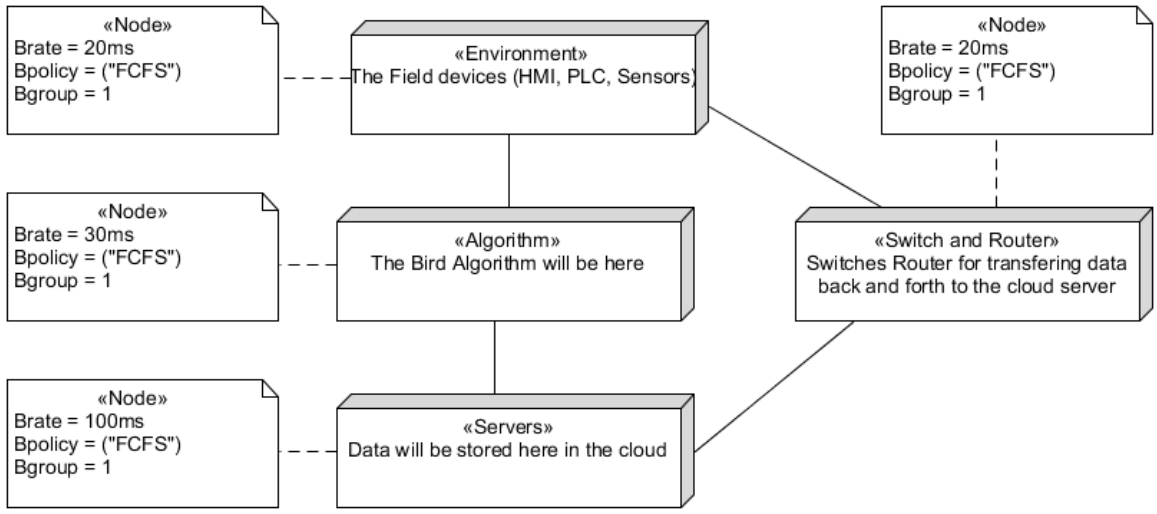


Figure 7-7 Annotated Deployment Diagram

With the information gathered so far, one can see the light from the tunnel. The efficiency of the system can be measured as well as the performance. For example, what is the cause of the rise in the time it took the system to actually store the data? From the start till the data was analysed was below 40ms and I believe it is very good for the system. from this deployment diagram, further activities are required for the movements of the information. This will determine the direction and activities information from environment will go through until it gets to the server. The question for example here will be, does every information or packets from the environment go through the network switch? The activity can be visualized by using the activity diagram based on the deployment diagram above. Figure 7.8 is the activity diagram based on the deployment diagram in figure 7.7.

There are four activities in the diagram starting from activity 1 to activity 4. The activity one is from the environment to activity 2 which is the network switch. The probability in the diagram named as “Bprob” is 1 which shows that any activity from the environment goes through the switch to the cloud. These are two ways information channel, any activity from the environment must go through the switch and the probability is also 1. The switch routes the information to the destination. Hence, information from the algorithm in the cloud can only be accessed through the switch from the environment. There is no access from activity 1 to activity 3, but there is an access from activity 3 to activity 1. The same way, the algorithm does not have access to the cloud, but the cloud has access to the algorithm.

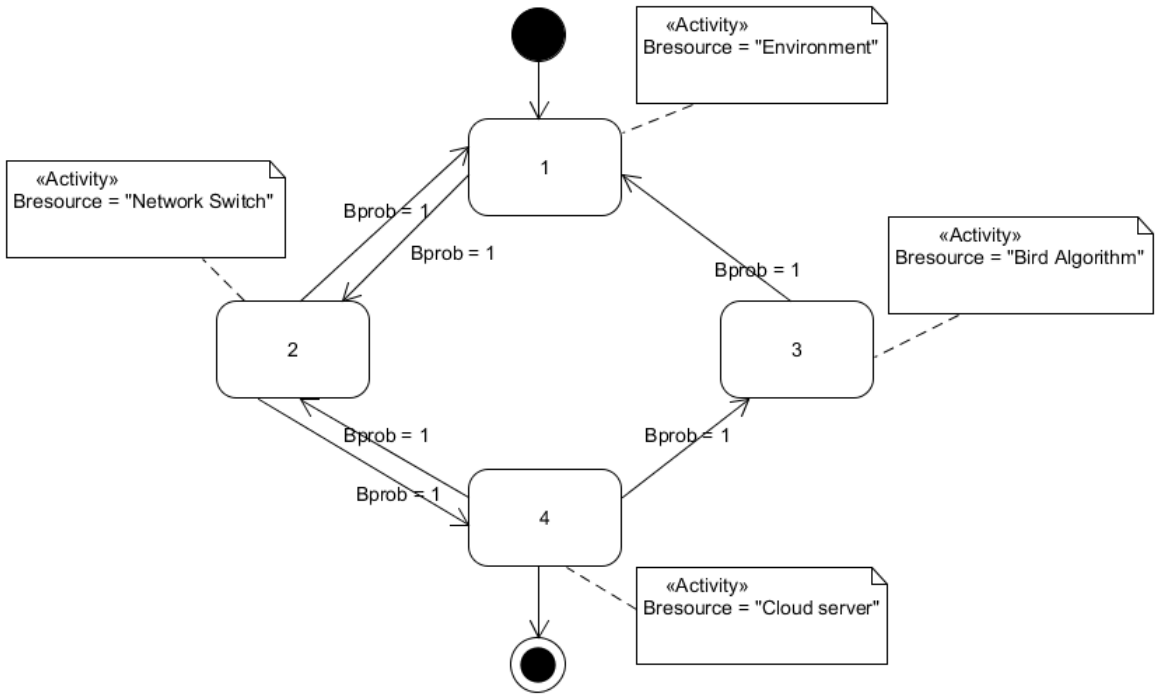


Figure 7-8 Annotated Activity Diagram with Transition Probability

7.3 Description of the Queueing Network Model.

The description in table 7.1 is the nomenclature that describes some of the letters and the meaning of mathematical symbols that will be used here. Queueing network as a network of ques was selected to be used here for the performance analyses of the model. The UML model has been transformed into annotated UML diagrams as seen above for better description of the approach. The diagrams are the activity diagram of the birds, activity diagram of the network and the service stations. The deployment diagram as well as the activity diagram from the deployment diagram were to capture the flow of information in form of transition and states.

The QN model as seen in the summary of table 7.1 is made up of some parameters. Firstly, there are some parameters in figure 7.2 which is the annotated use case diagram. These parameters are used to describe or determine some operation and make some informed decisions. The decision on the amount or the number of groups of birds that is needed to have in the system. The decision to know the better place for the location of the algorithm. This can be achieved by knowing the delays in the system such as knowing the arrival time of the protocols as well as the arrival rate. The service rate at which it takes the protocol from the start to land at the HBase and be saved in the server. C in table 7.1 represents the number of protocol while K represents the number of sensors. P is the probability of travel from one point to the other point. Please see table 7.1 for the rest of the information.

M/M/C	Queue Notation
C	Number of protocols
K	Number of field devices (sensors)
$\propto (t) :=$	Number of protocols arrival by time t
$\beta(t) :=$	Number of protocols saved at time t
rt_i	Arrival time of ith protocol
P = $P[i,r,j,s]$	Probability of a protocol r completing its service in the environment i and saved as s in the environment j (server) $(1 \leq i,j \leq K, 1 \leq r,s \leq C)$
$\mu = \mu\ [i,r]$	Service rate of protocol r on the environment i
$\lambda = \lambda[r]$	(Open QN) Rate of arrival of protocols r
q = $q[i,r]$	(Open QN) the arrival time of protocol r arriving on resources i
	detection to take place in the device i
N(t)	Number of protocol in the system at time t

Table 7-1 Symbols and Notation in the Algorithms

7.3.1 The Performance Measures

In queueing theory there are some performance measures that have been proved already which one can just plug in and start using. The theory of queues has been there since the early 20th century based on the work of Krarup, Erlang, 1909, reported by Thomopoulos, (2012). He used the approach to gather information on the telephone requirement. This help in measuring what is required for the efficiency of the system as well as best utilization of the system. Below are some of the approaches that have been proved, they might not be applicable in this system, but it what mentioning.

1. Mean System Size is the calculation of the mean value of the size of the system that is being measured. $\frac{(1+k)\lambda^2}{2k\,(1-\frac{\lambda}{\mu})\mu^2} + \frac{\lambda}{\mu}$ (1)
2. Mean System time $\frac{(1+k)\lambda}{2k\,(1-\frac{\lambda}{\mu})\mu^2} + \frac{1}{\mu}$ (2)
3. Mean Queue Size $\frac{(1+k)\lambda^2}{2k\left(1-\frac{\lambda}{\mu}\right)\mu^2}$ (3)
4. Mean Queue Time $\frac{(1+k)\lambda}{2k\,(1-\frac{\lambda}{\mu})\mu^2}$ (4)

7.3.2 From UML Model to Queueing Network Performance Model

Figure 7.1 to figure 7.8 are both the annotated UML models that described the architecture of both the software and the system. The Use case diagram in figure 7.1 was used to gather information on the architecture and workload of both the software and the environment. There are 13 use cases which some of them were merged to form one activity. The use case that described the detection approach by dividing the protocol and examining them were merged by using fork for the entrance and the exit in the activity diagram in figure 7.4. Actions in the use case and activity diagrams were stereotyped as “Bstep”. This is a process the system will go through in order to get to the next state or level. The location or place the process will take place was stereotyped “Bhost” and the action or services demanded from it was stereotyped “Bdemand”. These were to breakdown the activities as well as workload into smaller parts. The “Bdemand” is the request for the resources from the system by the software such as detect, which has to be measured with time. The important parameter here are the time and the requested resources. Below is the QN algorithm for the measurement of the arrival of the protocol in order to determine the required amount of bird that is needed as well as the system capacity. However, the “t” in the arrival rate definition is not the same thing as the “t” in the routing matrix. The “t” in the routing matrix is the transition, while in the arrival rate, the “t” is time.

Algorithm 1 For QN Generation

This is an open load system as the system received from the outside and goes outside also. There is no need for close system for the algorithm at this point.

{*initializations}*
For all *resources* $r \in R$ **do**
$count\ [r] := 0$
End for

For all $a \in A$ **do**
$count[res[a]] := count\ [res[a]] + 1$
$id\ [a] := count[res[a]]$
End for

$C := max_{r \in R}\ \{count[r]\}$ {Number of protocols}
$K := |R|$ {Number of resources}
P = $P[i, r, j, s] :=$ new vector [K x C x K x C] {probability of being saved}
$\lambda = \lambda[r] :=$ new vector [C] {arrival rate of the protocols}
$\mu = \mu\ [i, r] :=$ new vector [K x C] {service rate}
{Define the routing matrix}
For all $t = (x,\ y) \in T$ **do** **# t** = the transition which can either be [0,1]
$i := Bhost[x]$
$r := proto[x]$
$j := Bhost\ [y]$
$s := proto\ [y]$
P = $P[i, r, j, s] := p[t]$
End for
{Define the arrival rate of the protocols} ($T = (t_1, t_2, t_3 \ldots\ldots t_n)$)
For all $t \in T$ **do**
$i := Bhost[x]$
$r := proto[x]$
$\lambda[r] := \propto (t)$/t
End for

{Define the service rate of the environment i and j}
For all $a \in A$ **do**
$i := Bhost[x]$
$r := proto[x]$
$\mu\ [i, r] := \beta(t)$/t ## (N(t) - rt_i) *total number of time minus a particular length of time*

{Define system Capacity}
For all *resources* $r \in R$ **do**
$i := Bhost[x]$
$r := proto[x]$
If arrival rate ≤ service rate
Capacity = ∞
Else
Raise alarm for the overflow.
End if

End for

Algorithm 13 QN Generation

Simplified Version of Algorithm 1for QN Generation

{initializations}
For all the resources as seen in the deployment diagram $r \in R$ **do** **##($r_1, r_2, r_3, \ldots\ldots r_n$)**
Set the total count of numbers of action to the resource r = 0.
End for

For all the actions seen in the activity diagram on the resources $a \in A$ **do** **##($a_1, a_2, a_3, \ldots\ldots a_n$)**
count the actions on the resources
get the resource ID such as sensor ID
End for

$C :=$ count of the maximum number of actions on a resource
$K :=$ the number of resources
$P :=$ $P[i, r, j, s]$Probability of the protocol from the initial resource being save in the cloud = [K x C x K x C]
$\lambda =$ $\lambda[r]$ arrival rate of the protocols to the resources = [C]
$\mu = \mu\,[i, r]$ service rate of the protocol to the resources = [K x C]

##determining the routing matrix for the algorithm. From the environment to the cloud and back using the x and y as direction for both resources and the protocols to and from the resources

For all transitions x and y in transition T **do**
$i := resources\ [x]$ *##from resources i*
$r :=$ $protocol\ [x]$ *## protocol from resources i*
$j := resources\ [y]$ *##to resources j*
$s := protocol\ [y]$ *##protocol confirmed saved*
Probability of the protocol arriving at destination = $p[t]$ *## either 0 or 1*
End for

##define the arrival rate of the protocols from the environment to the cloud. T used here refers to time ($T = (t_1, t_2, t_3 \ldots\ldots t_n)$)

For all $t \in T$ **do**
get the resource
get the protocol
dividing it by the number of protocol on a given length of time will give the arrival rate
End for
##define service rate of the environment i and j and a here are the actions on the environment
For all $a \in A$ **do**
get the resource
get the protocol
dividing the total time by the total protocol at a given time by the length of time
##define system capacity for determining the possibility of buffer overflow
For all *resources* $r \in R$ **do**
$i := Bhost[x]$
$r := proto[x]$
If arrival rate ≤ service rate
Capacity = ∞
Else
Raise alarm for the overflow.
End if
End for

Algorithm 14 Simplified Version of QN Generation

With the information in the algorithm, one can simulate the measurements such as the arrival rate and service rate as well as the capacity. The information generated in chapter 4 of this work will be used. Based on the calculation, the average time it takes a packet to arrives was 20 and, in the cloud, was in the approximation of 100 milliseconds. With these data, for one server and the arrival rate of 5 as well as the service rate of 50. The arrival rate in chapter 4 was 4.6 and approximate it to 5 packets per second. The service rate was approximately 20 milliseconds and that should come up to 50 packets in a second. This information can be used to compute the average times as well as the waiting time and system utilisation. Table 7.2 and figure 7.9 are the simulated information using simulation engine Supositorio (Garay, 2014).

Table 7.1 showed that with the arrival rate of 28 and service rate of 50 for 1 server, the average waiting time is 25ms. The average time spent in the system was 45ms. This is the time it takes each to arrive at the destination. Since this is still below 100ms, it is still suitable and more can be added and simulated. By the way things are, since the service rate μ is greater than the arrival rate λ, the time spent in the system will still be less than 100ms. The service utilisation or the utilisation of the resources for the arrival rate of 28 was 0.56 which is still below 1. This shows that with one server, the system can take more data as far as the service rate is greater than the arrival rate as seen on table 7.2. Close attention should also be paid to the server utilisation which is still below 1 for the arrival rate of 28.

index	λ	μ	Number of Servers	Average time spent in the system (s)	Average waiting time (s)	Server utilisation (s)
1	5	50	1	0.0222	0.0022	0.1
2	6	50	1	0.0227	0.0027	0.12
3	8	50	1	0.0238	0.0038	0.16
4	10	50	1	0.025	0.005	0.2
5	12	50	1	0.0263	0.0065	0.24
6	14	50	1	0.0278	0.0078	0.28
7	18	50	1	0.0313	0.0113	0.36
8	20	50	1	0.0333	0.0133	0.4
9	22	50	1	0.0357	0.0157	0.44

10	24	50	1	0.0385	0.0185	0.48
11	25	50	1	0.04	0.02	0.5
12	26	50	1	0.0417	0.0217	0.52
13	28	50	1	0.0455	0.0255	0.56

Table 7-2 simulated information M/M/1 Queue

The graphs in figure 7.8 are the discrete probabilities as well as the probability-based time for the M/M/1 queue. The graphs on the left displayed both the accumulated and discrete information based on the number of customer and the probability. The right graphs show both probability of the time in the system and the time spend in the queue. The graph and the histogram on the left side shows the probability that there is n customer in the system.

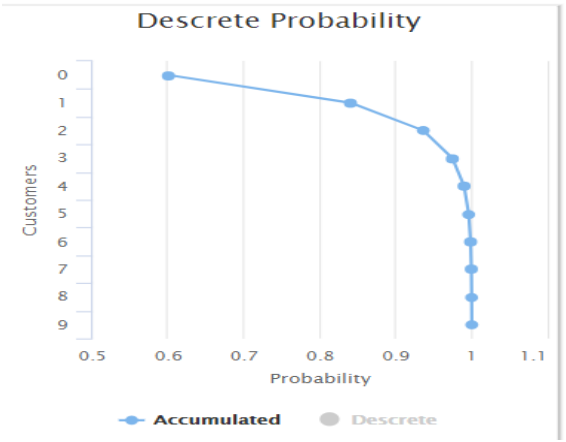


Figure 7-8a Discrete Probability

(for Accumulated)

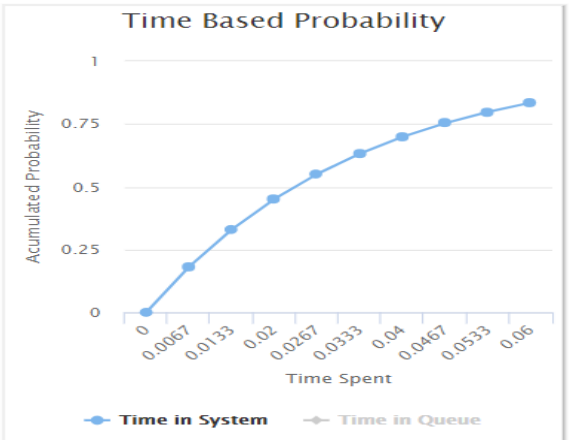


Figure 7.8b Time Based Probability

(for Time in the System)

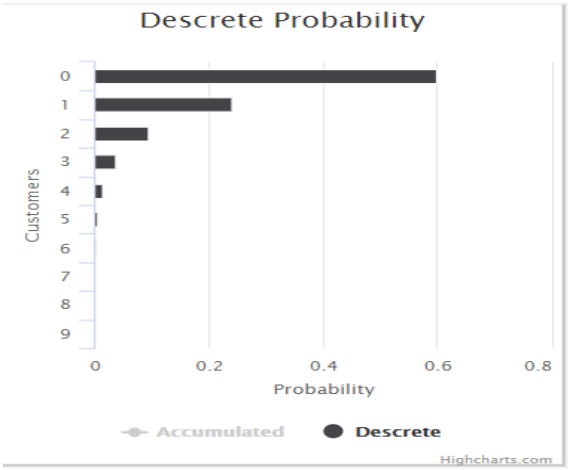


Figure 7-8c Discrete Probability

(for Discrete)

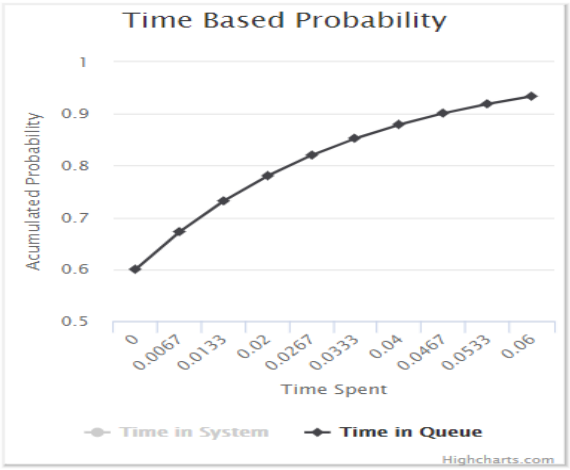


Figure 7.8d Time Based Probability

(for Time in the Queue)

The graphs on the right side indicated the time in the queue as well as the time spent on the system. These graphs will determine where the breaking point in making a decision for more resources. The probability of having a 0 customer in the system is around 0.6. The same goes to the time spent in the system as well as the average customer in the system and queue. The capacity of the system can also be determined. The capacity of 40% of average arrival rate of 20 and service rate of 50 is still fine. However, increasing the field devices without increasing the server will surely result in demand for more capacity or system utilisation. With the above information, an informed decision can be made with regards to the number of birds that is needed as well as determining and forecasting a denial of service attack. This can be forecasted based on the number of arrival and the service rate. When the arrival rate is equal to service rate, it is a danger on a yellow colour like the traffic light. However, when the arrival rate is greater than the service rate, the queue will be formed and the exponential increase of it will cause overflow.

The protocol arrival rate as well as the service rate was a straight-line QN. This means that there is no branching and feedback for the system. When the packet arrives in the algorithm with a command either to open or close, it is a straight forward approach. The packet is saved whether good or bad. However, with the algorithm, service channels can be more based on the number of groups available in the system. These are the groups of birds that resides in the cell and each cell can only accommodate a maximum of six birds. For a cell to qualify as a complete group, the number of birds there must be six. Hence, at the arrival rate of 1000000 per second, one server will not be enough. Based on table 7.2, the arrival rate of 20 with service rate of 50 will result in system utilization of 40%. Based on this information, 20 will be used to divide 1000000. The result will be 50000 groups or servers will be needed for such an arrival rate. Long queue will be avoided and longer waiting time will also be avoided. This determines the number of groups as well the number of birds that should be added to the system at any particular point in time.

7.4 Evaluate the Case Study Experimental Environment

The case study for this project was carried out as described in chapter 5 and 6. Train or locomotive system was used as a test rig for the model. The locomotive system was built with some of the field devices such as PLCs and HMIs as well as other sensors. The model that was used in chapter 6 was explained in chapter 5. Hence, evaluating this will be partially evaluating the model. The case study was actually the application of the model. The models here

demonstrated on the train system was about organising the movements as well as the track operation. The modern track system is controlled from the control room and some of the systems are automated as well. the track system used in chapter 6 demonstrated how accident could occur if not manned by technology such as AL or person in the control room. The weakest link in the security circle is still the human being as explained by Schneier, (2005).

The planning was to gather all the information that was needed about the system. The track system was mapped out by allocating numbers to the sensors as the transition and the place before and after the sensor for locating the position of the train. Each sensor has a neighbouring sensors and places. All the field devices are regarded as sensors and every sensor is regarded as bird. These sensors belong to a group or groups and communicate with each other in the group and groups. These was to demonstrates the intercommunication between groups. This intercommunication between groups was one of the approaches the bird in the group use to determine danger in the system. Birds such as PLCs and HMIs do not move but the programmed birds with ID starting with B changes groups as they move from group to group. This type of bird is allocated to the sensors or the group for anomaly detection. They make decision based on the available information, either from their group or other groups. The sensors have infinite TTL as well as some birds that move. The one that do not have infinite TTL, die or being removed.

The evaluation of this type of bird or the model is to determine its usefulness in the system as a decision maker. Any wrong decision by these birds will end up in a catastrophe such as crash or collision in the system. The modelling tool that was used is CPN from Ratzer et al (2003). The analysis and evaluation will be based on the capability of the tool and some other skills. The result will be to test the queue and the movement of the locomotives in the system. Determining a deadlock in the system by the birds as well as other properties. The first model to evaluate will be the TP1 model of the sensor. This is a demonstration of locomotives coming from three different locations and heading to a single location. The AL supposed to detect such an anomaly early on the system. Below will show how fast such anomaly on the system will be detected.

Using the assumption from the QN model of the detection or operation as shown in figure 5.8, which was a maximum of 20 milliseconds. For a software operation in this range is assumed very good. The model that was used in the train system for securing the flow of the system,

used queues. However, anomalies are checked in every transition and places before a transition occurs. Figures 5.3 and 5.4 showed that a bird can go from scanning to idling as well as from idling to scanning. It is either that a bird is scanning, or it is idling, and the scanning birds are the ones that were used to solve the detection of anomalies. The idling birds are in order words, in a disabled mode while the scanning birds are in enabled mode. The scanning birds can spawn others as shown in figure 5.5. That means they reactivated the disabled birds which were disabled as a result of inactivity within their Time-To-Live. Birds are in groups and groups share information among other groups as a bird can belong to more than one group. These was the approach that was used in the train or locomotive system. The train track has some areas that were mapped out as risk locations and should be monitored. The monitoring was organised in such a way that the connections in that area belongs to the same group. This includes the PLCs, HMIs and other sensors. These groups have to communicate with each other for the protection of the system. This communication is between birds that move from one group to the other. However, they check the state of the sensors, open or close based on the information received, informed decision will be made. This decision will tell whether there is an anomaly in the system or not. The main aim of this project is the detection and based on that, anomaly can be predicted based on the location of trains and the state of the sensors.

The performance of the simple queueing system in figure 6.5 was evaluated. As stated by Wells (2006), performance is measured by the way workloads are being processed. Her analysis and introduction of the performance measure shows that CPN Tool can be used for effective evaluation of performance. Hence, figure 6.5 was evaluated on the processing of the queue and data was collected. The transition “Comm” was timed initially for 10 unit of time and the simulations were carried out for 300 steps, 150 steps and 30 steps. Each token colour set was increased to 10. The three colour sets were 30 in total, representing three different sets of trains and 30 trains in the locations. The initial configuration was 3 different trains and the total number of trains were 3. There were also 3 transitions as well as 5 places represented in the model. However, the transitions represent the activities from a place to a place. These transitions can be enabled based on the configuration of both the places and the transitions. A queue system was dimmed right for this train system because the workload described here is the train. Train can collide or formed a deadlock if there is no way of detecting such anomalies. In order to avoid such situation, queue system that is being monitored by our bird (AL) model was modelled. With the transition “Comm” timed for 10 unit of time and 30 trains runs for a

total of 90 steps at time 300. Below are some of the data collected when comparing the outcomes of the simulation. For the simulation results, see appendix, pages 38 to 49

7.4.1 Data from the Simulation

Comm Transition Data Collection at time 10 unit

Number of tokens = 30

Tokens colours = 10`Train2@0 ++ 10`Train1@5 ++ 10`Train3@10

Number of Steps = 90

Time = 300

Maximum number of trains in the queue = 27

Categories and number of trains in the queue (62) = Train 2 = 8, Train 1= 10 and Train 3 = 9

Steps with the highest queue = 62 and 63. Step 63= Train 2 = 7, Train 1= 10 and Train 3 = 10

Comm Transition Data Collection at time 5 unit

Number of tokens = 30

Tokens colours = 10`Train2@0 ++ 10`Train1@5 ++ 10`Train3@10

Number of Steps = 90

Time = 150

Maximum number of trains in the queue = 26

Categories and number of trains in the queue (63) = Train 2 = 7, Train 1= 10 and Train 3 = 9

Steps with the highest queue = 63 and 63. Step 64= Train 2 = 6, Train 1= 10 and Train 3 = 10

Comm Transition Data Collection at time 1 unit

Number of tokens = 30

Tokens colours = 10`Train2@0 ++ 10`Train1@5 ++ 10`Train3@10

Number of Steps = 90

Time = 30

Maximum number of trains in the queue = 18

Categories and number of trains in the queue (71) = Train 1= 9 and Train 3 = 9

Steps with the highest queue = 71 and 71. Step 63= Train 1= 8 and Train 3 = 10

At this time, the maximum queues were also in steps 21 and 22 which has T2 = 8 at step 22

7.4.2 More Results and Further Analysis

The timed colour sets 10`Train2@0 ++ 10`Train1@5 ++ 10`Train3@10 were reduced to 10`Train2@0 ++ 10`Train1@1 ++ 10`Train3@2 for the same 10 units of time at Comm. The results were different, the queue increased from 27 to 29. Train 2 were 9 while others were 10 trains for each colour set. The locations of the maximum queues were in steps 61 and 62. The timed "Comm" transition was reduced to 5 with the same colour sets. The maximum number of trains in the queue increases from 26 to 28 with the highest number of queues found in steps number 61 and 62. The same procedure was carried out with by only changing the "Comm" transition time to 1 unit of time. The queue increased from 18 to 26 and the location of the highest queue was in step number 63 and 64. Train2 was 7, Train1 was 10 and Train3 was 9 trains on the colour types. However, with the same "Comm" transition set to 0 unit of time, the highest number of Train in the queue was 4. The steps were the same for all, but the time changes. Also reducing each colour set time to Train2@0, Train1@1 and Train3@2 with "Comm" transition set to 0 unit of time, the highest number in the queue was 3. The step was the same while the time was 2.

7.4.3 The Queues and Time of Service

From the information and analysis collected so far, one can point out some facts based on this information. Figures 6.10 to 6.14 dealt with the formation of groups in the system. The sensors are grouped together with birds for monitoring the sensors. These sensors are regarded as birds, but the difference is that they do not move and do not have limited TTL. Their operations are also limited to their groups. These formed the models and was used for the Train system. The model organised the sensors into groups as seen in chapter 6 figures 6.10 to 6.14. These models have transitions and places. The transitions are for sensors, while the places hold information that determines the state of the transitions and location of train if there is need for it. In order to determine the effectiveness of the model in terms of performance, the queue in figure 7.9 was used.

The information in section 7.3.1 and 7.3.2 were from the simulation carried out on the queueing model. As earlier described, queues can be used to organise a system as well as a way of avoiding deadlock in the system. Although our system is distributed, many groups can work in parallel. One thing that stood out in the experiment was the steps from the beginning of the simulation to the end of it. The steps remain constant while the time changes and this is an indication that it is not dependent on the time. Steps are dependent of the transitions that are

being monitored. When a transition is enabled, the step is counted and recorded. Figure 7.9 has three transitions and the number of colours were 30. Each colour set has to move three steps, that will make it 90 steps. The queue concentration was on the different levels based on the delay time on “Comm” transition. Reduction on the time there resulted in lowering the number in the queue. This shows that the reduction in the service rate will reduce the queue just as in the queueing theory. The performance of the system is based on the number of Trains in the queue which determines the rate of service. Reducing the service rate has proved effective by reducing the time of the transition. Another observation from the simulation was the service approach which was First Come First Serve (FCFS). Throughout the simulation, Train2 was programmed with zero time and others have time attached to them. Hence, the Train2 was first processed before any other train in the queue. The number of transitions adds to the service rate as seen from the steps and the time in figure 7.9. It has three transitions which shows three processes. This also represents three stages of operation and each stage has its own time. Therefore, reducing the number of processes will definitely increase the service rate and reduce the queues.

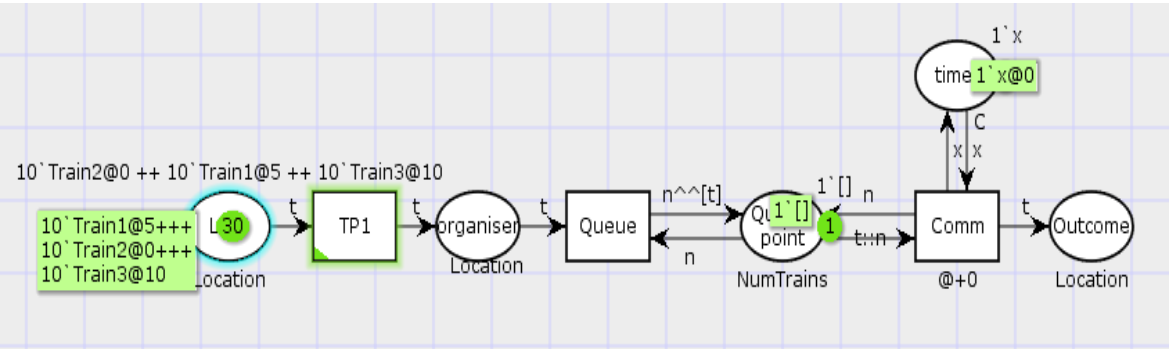


Figure 7-9 The Queues

Figure 6.10 has two transitions and four places as seen in chapter 6. There are two birds with the unique ID with B. The group has to deal with two transitions and record information on the places. As the policy is FCFS, the birds will definitely detect queue before arriving at their first place from P2, P13 and P14. These are three trains coming from different locations and heading toward TP1 sensor. This is a complete deadlock and should be detected, which was modelled as queue in figure 7.9. This formed the model from figure 6.3 and the best alternative was to detect it earlier than now. Figure 7.9 started with the location to TP1 transition. It represents the trains from different locations. Time was modelled into the colour set as a way of scheduling the trains. The queue is formed by the B1 in figure 6.10 which represents the queue in figure 7.9. The place B2 is represented in figure 6.10 as place queue point. The transition “Comm” forms the extra measure that is put in place for checking and verification, which can

be removed if it is causing too much overhead with regards to time. Figure 6.10 shows model as a scenario with three trains which has the possibility of deadlock. The solution for it was given based on the queue approach in figure 7.9. The trains can be allowed to queue and will be served later based on the direction, FCFS and deadlock avoidance. The same queueing approach can be applied to all the groups from figure 6.10 to 6.14 and the performance will be as explained in section 7.3.3. The queues increase as the transition and the time of service increases. If the time of service is reduced by reducing the number of transitions, the systems and the software will work very effective and efficient. One of the things that can influence the service time is the hardware the system is using. Therefore, evaluating the hardware that was used here is vital and the next section will be the hardware evaluation.

7.5 Hardware used

Three types of systems were used during data collection and it is good to mention that each of them has its effect on the system functionality and efficiency. The operation of systems does not only depend on the type of software that is stored on the system, hardware has its part to play. These three systems will be categorised as system 1, system2 and system3 as seen below.

7.5.1 Systems 1, 2 and 3

The hardware specifications of the first system are listed below.

- Operating system: windows 7 64 bit
- Random Access Memory: 20GM
- Central Processing Unit: Intel® Xeon CPU E5-1620 3.6GHZ
- Storage Capacity: 1TB

The hardware specifications of the second system are listed below.

- Operating System: CentOS
- Random Access Memory:4GB SO-DIMM
- Central Processing Unit: 16 x Intel® Atom-Core E3845 1.9GHZ
- Storage capacity 16 x 66GB

The hardware specifications of the third system are listed below.

- Operating System: CentOS
- Random Access Memory:141GB
- Central Processing Unit: 16 x Intel® Xenon CPU E5620 2.4GHZ
- Storage capacity 80TB

The first computer is the engineering workstation which was connected to the environment via switch. There was no Cloudera on this system, it only has the configuration of the SCADA. The train system equipment such as PLC and HMI were configured in this system. Wireshark software was installed also for data collection. Data was collected with the Wireshark software as the system was running. The average time of the protocols was measured, and it came up to about 20 milliseconds. The second system was a master slave system with Cloudera installed in it. It has 16 slaves or nodes. The system is connected via network switch to the master. Apache flume was configured to ingest data to the HBase and HDFS. The configuration will sink apache flume with the HBase and create channel and table as well as column family for that. There is a serializer that comes with the flume which will convert the flume event to HBase format. However, the rate at which the data arrived in this second system was not far different from the first one. The arrival time was between 24-32 milliseconds which might be due to the travel time. The third one was a server which was configured with Cloudera and HBase. The system was connected with router and switch. The router was for wifi and two Raspberry PI with USB port for wireless access. The Ingester was written that channels data to the server, HBase. The Data was collected, and the time was a bit more than others. The average time of the data was between 40 and 89 milliseconds.

The data collected from the three systems was a proof that it is possible to keep the algorithm in the cloud. This resolves the issue of big data and latency and because 40 to 100 milliseconds are still very good time (Jovic and Hauswirth, 2010). Hence, the possibility and viability of the approach is true. Maximum results can still be achieved even when the detection algorithm is placed in the cloud.

7.6 Algorithm

The algorithm has two parts as seen in the later part of chapter 4 and chapter 5. The model described in chapter 4 was modelled for the protocols used in this project. The main protocols here are ProfiNet or S7 and Modbus protocols. These two protocols are the widely used protocols in industrial control system and SCADA system in particular. The simulation of the bird's algorithm with NetLogo programming and simulation platform yielded a very fast detection (Wilensky, 2016). This was the average time it took the group to detect the predator in the environment. The time in this environment depends on how the procedure was set for the flock to run. The timer was set with NetLogo variable called LogoTime. Patches were setup as part of the environment with capacity of each patch set to 6. The speed of the flock was

based on the slide and can be adjusted. However, predator can be added to the environment and the flocks are meant to detect the predator. Based on the fact that birds are flying all over the environment, from patch to patch, detection was instant. Immediately a predator is added to the environment, it only takes between 0.01036 to 0.01318 seconds to detect it by the groups. This shows the average time of between 10 to 13 milliseconds. This is very fast and suitable for the purpose. Although the system in which the algorithm will be running from has more processing capacity. The processing power and storage capacity will not be a problem for this system. However, testing the algorithm in a more lesser processing power is also vital in order to know pros and cons of the algorithm.

7.6.1 Generated Dataset

The two datasets that was generated are the Modbus and ProfiNet protocols. The fact that this project tested the system with a real dataset that was generated is a plus. Testing and analysing in a SCADA environment can be very crucial based on the fact that any disruption can be very dangerous. Two different datasets were collected, and a model was constructed based on this data. This provided us with the vital information on each dataset. The Modbus dataset was an easy dataset because there was model from Goldenberg and Wood (2013). The Model of the ProfiNet or S7 dataset was there, which is from Kleinmann and Wood (2014). However, the S7 dataset model was for the old model of PLC with protocol ID of 32. The newer model of the protocol has a different ID, 72. With the idea and a careful study of both models and the protocols, the required information was fished out of the datasets. This information was used as the initial experimentation and knowledge acquisition of the workings of these protocols. This information equipped us with the knowledge on how this information is being packaged and transferred practically. Having a system that works with two different protocols is a sign that more can be done to accommodate all kinds of protocols. Although a system such as Open Platform Communication (OPC) can work with different types of protocols but not yet for IDS. This has given this work its uniqueness by combining these two protocols together and finding a way of using them for IDS.

7.7 Methodology Used in the System

Design Science (DS) methodology was used for this project as described in appendix. A case study was used as a method, or the approach to describe the operation of the model. Although the product of this work is a model representing the solution for a SCADA system and ICS. The DS methodology that was chosen for this work was appropriate because it was a guide to

this point. The DS methodology mainly focused on problem solving in form of an artifact. That means, solutions are represented as an artifact. According to Havner et al (2004), design is a process of discovering the best solution through search. It also shows that artifact can be in form of; Construct, Models, Design theories, Instantiation and Methods. This work has chosen to represent the outcome as a model and models are mainly a way of representing ideas based on symbols and vocabularies. The methodology has guided the project both in recognising the subject matter of the project as well as following the timeline in finishing the project. It recognises the strength and the weakness of the project such as stated early in the methodology. Hence, following the plans, the initial questions that were translated into research objectives were answered. Models was developed, and case study was constructed and evaluated. Knowledge has been created through the creation of artifacts as the primary goal of the methodology. It is quite well understood that many in the technical field would suggest and ask, why only model? It has been demonstrated that a project that drives for many years are the ones that have a proper foundation. Vaishanav and Kuechler, (2007) pointed out a successful PhD project that produced models and the idea is still alive and well today. The reason for the drive was because of the proper foundation and each successive student build on it. This can be seen in most of the big software available today, the power of corroboration. The idea may start with one person and putting it on paper will make it firmer. This will give the next person that will work on it a head starts. The two models produced in this work following the DS methodology was a success. However, that does not mean that producing models are the easier option to a project of sort. There were problems encountered during the project life circle which will be explained next.

7.7.1 Any Discovered Issues?

There were issues from the very start of the project but that did not stop or deter project idea. The first one was during the introduction of the project. The scope of the project has not been defined and the understanding of the scope was actually a problem. The title of the project was "Artificial Life as a Vehicle in Detecting Malicious Behaviour on ICS". The definition of artificial life was uncertain as well as malicious behaviours and ICS. Artificial life can be any life that can be emulated. Malicious according to Cambridge dictionary means an intention to harm or cause harm. This has a wider definition in security as it can involve anything that operates within and outside a system, knowingly or unknowingly. This was filtered down to abnormal behaviour of the system in order to make the project feasible. The next was on chosen the methodology that will be used in achieving the aim and objectives of the study. The

methodology was carefully chosen after examining literatures and the scope of the work. Based on the fact that this work is first of its kind, a strong foundation is needed so that anyone that will continue with it will have a base to start. Modelling these approaches wasn't as easy as it looks. In order to model something, enough background information is needed. These are the information that will be fed in to the model. However, the initial stage was on what to model, the bird or the environment. Flocks or bird without the environment cannot work because birds live in the environment. This goes onto the definition of the case study as well as data collection. Establishing what is to be modelled can be an issue if the scope is too big. This was the reason there was more than one model represented here. The first one was for the protocol and the last one that was used as a case study was for the environment. However, the initial confusion brought about the robustness of the work by creating extra model.

The analysis and evaluation of the model posed issues of its own. The first question was, how do you evaluate a model of this kind? After some review of literature on the model performance evaluation, QN model was adopted. Although there were issues encountered, but they gave us the opportunity to think and make decisions. The results of all the decision and work was model that can be implemented on SCADA system and ICS in general.

7.7.2 Discussion

The process has been a journey that worth following. As said by Philip Stanhope, "Whatever is worth doing at all is worth doing well". This is one of the reasons for chosen models and every effort was to get all the needed information that can help anyone drive the idea further. The model was a contribution to knowledge and the objectives of the study were met. More would have been done but the time for this project was three years and one can only achieve much within this time. Understanding the project can take years as well as its feasibility. The evaluation was not all that straight forward compared to evaluating a software application. This was an evaluation of a model that has been designed. Some tools have been designed for performance evaluation such as CPN Tools that was used for modelling in these work and other tools. However, in order to get more out of a model, every step has to be analysed and evaluated. The same way the analysis was carried out in chapter 6 for all parts of the train system. Things to remember are the questions that were asked in the methodology (see appendix) that formed the objectives of this project. The questions were as follows;

1. How would the detection approach seen in the flocks of bird be modelled for anomalies detection on ICS? This question was answered in chapter four and chapter five of this project. In these chapters are the model representation of the flock of birds and their detection strategies for ICS.
2. How can this approach solve the problem of huge amount of data being generated from sensors? The problem of big data was solved by integrating the system in the cloud and Hadoop framework. The benefit of cloud computing was explored, and the analysis showed that it is possible to use cloud and Hadoop for data storage and algorithm.
3. How can this model be compared to other detection approaches such as Snort rules and models such as Artificial Bee Colony (ABC) and Ant Colony optimisation (ACO)? Snort rule is a signature base detection engine which has the limitation of detecting only known anomalies. The current and future IDS approach such as the one presented in chapter four and five detects both known and unknown anomalies. Snort rules does not recognise the SCADA data while ours is specifically meant for the SCADA system. Comparing this to other approaches such as ABC and ACO, our approach is distributed in nature and any bird can detect. ABC and ACO has leader while our bird flock does not have leader, but they flock together and do corporate with each other.
4. Where in the SCADA systems can the model be applied? The model can be applied as seen in chapter 4. It can either act as a middleware or it is integrated in the cloud. If the algorithm is resource intensive, then cloud option will be used. This is a matter of choice and security for any company.
5. How would the anomalies be detected on SCADA systems data? Anomalies are anything outside the normal use of the system. Therefore, it can be internal or external and the detection here are based on the systems protocols as well as the system operation. Anything that can hamper the normal operation is deemed anomaly and must be detected.

With the answers to these five questions above, the objectives were met. However, the objectives and the hypotheses as well as the aim will be further scrutinised. The scrutiny will be to know whether the objectives were met and the aim of the project. Whether the hypothesis was followed throughout the life circle of the project.

Summary

This chapter started firstly with the evaluation of the UML model that was created in chapter 4. The model was first uploaded to Udemy platform for evaluation. Some of the colleagues about 38 in number in the UML class gave their opinion of the model. However, such evaluation was not taken into account as some of them did not understand the model based on their comments. After reviewing some literatures on the evaluation of UML models, performance was one of the key factors being evaluated. Hence, performance of the model was picked to evaluate the model against. In order to evaluate the performance of the model, an approach was adopted called QN model for performance evaluation. The UML model has to be translated to QN model in order to be evaluated. Algorithm was constructed that will convert the UML model to QN model by extracting the required parameters.

In order to know the required parameters, the use case diagram has to be revisited for the recognition of the workload. The use case shows the workload which was used to construct the activities of each work, which was represented as activity diagram. The activity diagram was a decomposition of the use case diagram. Both the use case and the activity diagrams were annotated in order to virtualise the activities and extract the parameters. There was annotated activity diagram for the bird, network and service station. They were all annotated for the performance evaluation of the algorithm. The QN model pointed to the data that was collected during data collection. This data was used to measure the performance of the system and helped in making informed decision on the system capacity as well as arrival rate and service rate.

The second part of the evaluation was the evaluation of other part of the system such as the hardware used. The effect each system equipment had on the performance of the system were recorded. However, not much difference was recorded but the functionalities were almost the same for all the system. The methodology used was evaluated and the outcome was that it yielded maximum result and was effective. Both parts of the evaluation suggested that the model is suitable for the purpose.

CHAPTER VIII

8 CONCLUSION, CONTRIBUTION AND RECOMMENDATIONS

The project started in the first chapter by introducing the Industrial Control Systems (ICS) which was filtered down to Supervisory Control and Data Acquisition (SCADA) as the focused environment for this study. The areas of implementation of SCADA system such as nuclear site, locomotive, aeronautics, Industries, Hospitals, Transportations systems and among others defines the significance of it. It shows the important of the system in our daily lives based on their area of applications. In the past, the system worked in isolation, but the present connectivity has put the system into numerous security questions. The connectivity has allowed the system to be monitored from anywhere in the world through the internet. The present generation of the system connectivity has put this vital system in danger of which any malfunction of the system can endanger lives (Yardley, 2008).

Chapter two and three dealt with both background information and literature review, both the current and past literatures in order to be informed of what has been done in resolving this issue. The review of literature highlighted both the present and future approach in detecting anomalies in such systems. These anomalies can be both internal as well as external and it can be intentional as well as unintentional. Anomaly is anomaly no matter where it is coming from, the only thing is to detect it before it causes damage. This is the reason for looking into the approach seen in the flocks of birds for anomalies detection. This animal moves in thousands and even million but capturing one from that million can be very difficult compared to a lone bird. This means that a lone bird is easier to capture than a bird in the flock. The most amazing thing here is that, each bird is capable of detecting a predator (Cavagna, 2010). They do not have a leader, but their corroboration is phenomenal. This approach was deemed necessary and vital and replicating it in SCADA system for protection will enhance anomaly detection. After the study of the available detection approaches, the bird's detection approach was like the hope of the modem detection. The methodology for achieving this goal was devised and chapter four started out with the system design and later the first UML model.

The main focus of this project was to produce a model of the artificial life that can be applied to the SCADA system for the detection of anomaly. This project used the approach of the bird flock to predator. That is the way prey detects predator when they are in the flock. Each bird

in the flock can detect predator and their detection approach was studied and modelled which is the product of this project.

8.1 Research Hypothesis Revisit

The whole of this project is about proving the hypothesis right or wrong. From the question about the security of SCADA system as a vital platform that lives depends on, come the hypothesis. The hypothesis is about hypothesising the best way that will improve the security of SCADA system. Below is the original hypothesis that this project was built on;

"*Predator prey model approach can enhance the detection of anomalies on SCADA industrial process control systems"*

Before the above hypothesis was devices, a lot of thought and research was devoted in understanding the area of application. The area or the platform that needed security is the ICS and in particular SCADA systems. The main focus of the project is on SCADA, but the approach can be applied to other areas of ICS. The findings of the research that was carried out as well as the experiments showed that security of SCADA system is paramount. The system is implemented in most strategic positions in the world and any deviation from the normal operation will surely endanger lives (Dickinson, 2013). Take for example the demonstration that was carried out in the school laboratory with the traffic lights. Traffic lights are not manned by anyone physically, what would happen if in a T junction, all the traffic lights turn green at the same time? This will surely result in catastrophe and lives will be affected. This kind of situation can be caused by system malfunctioning and before alerting the police, lives must have been affected. This is just the simplest one that was demonstrated that anyone can relate with. There is nuclear station connected, Locomotives, Aeroplanes in the air sending signals to the control room and many more. These are issues and concerns and the question on how best to secure this system, as many of them were built without security in mind. Exploring literatures and nature, the idea of flocks of bird came to light. They do not have leader and they can flock in thousands, do some aerodynamic movement in the air with excellent coordination. Each one of them in the flock is capable of detecting predator. Looking at what has been done in security for securing ICS, I believe that this is a big step into another realm of security. Surely, the approach can detect anomaly such as explained with the traffic light. This is one of the evidences that the predator prey model can enhance the detection of anomalies on SCADA industrial process control system.

8.2 Research Aim and Objectives Revisit

The primary aim of this study is to explore, experiment and explain through models the use of artificial life in detection of anomalies on ICS. The question now is, whether the aim of the project was met? The aim of the project was met and the artificial life here is the bird and their approach were modelled, which can be used in securing SCADA systems. The model showed that it is possible to use the flocks of bird's approach in detecting predator for securing SCADA systems (Okeke and Blyth, 2016). The objectives of this research are below;

1. To carry out research on the possibility of using the flocks of bird's approach for anomalies detection on ICS.
2. Carry out research on the possible best way to model the approach for the detection of anomalies on ICS
3. Carry out research on the best possible solution for big data
4. To create a SCADA test rig upon which a predator prey model can execute and to create a predator prey model. The test rig will facilitate the collection of initial data from the PLC and HMI that will be fed into the model in order to determine the anomalies.
5. To create the models and evaluate it for performance in order to validate the hypothesis.
6. To compare and contrast the result in order to prove or disprove the assertion that a bird of prey model can enhances the detection of anomalies on Industrial Control Systems.

8.2.1 Research objectives 1, 2 and 3 Accomplished or Not?

Chapter one and chapter two of this thesis were devoted to both introduction, background and literatures review. Chapter one introduces the subject of this research and chapter two the background, while chapter three went into review of literatures in these areas. However, it was evident in chapters two and three that the objective I and 2 were met coupled with the journal article from this research. The journal article was on adopting the flocks of bird's approach to predator in securing ICS (Okeke and Blyth, 2016). With this information, the objective number one was accomplished. The model produced in chapter 4 and 5 showed that the objective number two was also accomplished. Through the research that was carried out on the best possible methodology (see appendix) through to design in chapter 4, the models were produced. Understanding the structure of the flocks of birds helped in producing the model for the detection of anomalies. Further research on the current detection approach for big data also showed that using cloud computing and Hadoop will resolve the issue of big data. The distributed approach seen in the flocks of bird is a solution for the detection (Okeke and Blyth, 2017).

8.2.2 Research objective 4 Accomplished or Not?

To create a SCADA test rig for generating data that will be used to test the predator prey model. The test rig will facilitate the collection of initial data from the PLC and HMI that will be fed into the model in order to determine other possible sources of anomalies as well as model performance. The question to answer here is, was this objective accomplished? The answer here is obvious because the models were shown in chapter 4 and 5. The data collection was carried out in chapter 4. The model was evaluated based on performance using QN model in chapter 7. This showed that the objective number 4 was met.

8.2.3 Research Objective 5 Accomplished or Not?

To create the models and evaluate it for performance in order to validate the hypothesis. Models were created in chapter 4 and chapter 5 that validated this objective. The hypothesis stated that the model will enhance the detection of anomalies on SCADA system. Our analysis in chapter 6 and 7 showed that the model is suitable for SCADA system and can enhance detection of anomalies. The model is distributed in nature compared to other models. With this, the objective was accomplished. Therefore, that aspect of the objective was fulfilled.

8.2.4 Research Objective 6 Accomplished or Not?

To compare and contrast the results in order to prove or disprove the assertion that a bird of prey model can enhances the detection of anomalies on Industrial Control Systems.

Based on the result of the evaluation and analysis in chapter 7, it is obvious that the birds of Prey model can enhance the detection of anomalies on SCADA systems. Comparing the model with Snort rule and other detection algorithms such as Artificial Bee Colony and Ant Colony Optimisation. Snort rule is based on rules and it does not detect unknown anomalies. ABC and ACO might detect unknown but their detection approaches are limited. They have hierarchies and the detection can be based on hierarchy. Our birds of prey model do not have hierarchy, any bird in the flock can detect predator. This shows that the research objective was accomplished.

8.3 Future Works

This work laid the foundation of the greater work to be done in the area of anomalies detection on ICS in general. Although many researches are now focus on securing the system as the security community are now understanding the important of this system. Studies are now being carried out on the security of critical infrastructure. The increase attack on the system is causing the world many things such as money and most importantly lives as seen in train collision around the world. We have witnessed some recent attacks in Europe and America. Our

approach focuses more on the detection of anomalies and little attention was given to prevention of the attacks on the system. The future work will mainly focus on preventing attack on ICS as well as implementing the model.

8.3.1 Intrusion Prevention Using Birds of Prey Approach

Intrusion prevention that was modelled into this model can be refined and broadened to do more. The prevention can cause a button to be disabled if pressing it at that time can cause problem. Birds flocking together in thousands and even millions corroborate with each other in such a large group. They watch each other's movements and acts accordingly. The same way prevention can be strengthened through more communication of birds in the flock. This can be from the hardware down to the software in the system. This means that security can be integrated straight to hardware as well as software by more corroboration between these two groups. The same way the operating system in computer talks to the hardware, the same way security can be built. Software corroborating with the hardware to detect anomaly in the system and stop intrusion.

8.3.2 Predator Avoidance mechanisms

One of the most amazing things with birds in the flock based on the study carried out with the European starling is their manoeuvring in the air. Sometimes predators see them as easy target and to their surprise, hard to catch one. Predator can go through them but would not catch one until the predator is tired. This can be built into aircrafts for avoidance or even cars. This work can be taken further in these areas to help prevent accidents or even being hit in the air by missiles.

8.4 Conclusion

The rate at which the Industrial Control Systems are being attacked is a call for concern. The problem is that you do not need to be a professional hacker to penetrate into such system. Many software's are available online, that at the click of a button, you are into a system. This is like equipping an armature or children with weapons. YouTube videos are available on how to penetrate into systems. Although security personnel are working hard in bringing new ideas on how to secure such system, but they are still behind. The attackers are not sleeping, they are also working hard in exploiting new ways of penetrating into systems. We have heard in the news of many attacks and accidents that could have been avoided, such as train collisions, power system shutdown. Shodan online can show some of the connected systems with their IP addresses online. If lives are dependent on these systems, the time is now to wake up to the call (McMillen, 2016).

Most of the problem in the security or with securing a system is that many security strategies are very hard to implement. Because security systems are hard to implement, hackers use that avenue in introducing anomalies into the systems. They explore that vulnerability to their own advantage. The problem is that most people do not know the reason why they have to change their password every three months depending on the policy of the organisation. Since hackers new that the system is very secure, they would not try to exploit it through their traditional route, rather they use another simpler means. The most secure systems are mostly penetrated through the weakest link. The weakest link in the security circle is still the human. If security is simple and the users have good understanding of it, security would be easy. This would be like a second nature to every worker and system user (Vacca, 2017, chapter 1).

This is the reason for proposing our approach of using birds of prey model in detecting anomalies on SCADA systems. The approach is simple and can be implemented in a wide range of systems. The model in this project was mainly demonstrated with the SCADA system. However, the model is not limited to only SCADA system. For the purpose of this project and the demonstration of the idea, SCADA system was chosen as an area that needs urgent attention. The project has a good prospect for anyone even for further PhD work and masters, as well as undergraduate degree. The growing use of ICS in many areas of the society demands that this be studied as part of curriculum for students. The more people know about the system and the benefit of it, the more security is needed. The new era of security is being ushered in as the bird of prey is introduced. The performance measure adopted showed that, birds of prey

model can enhance the detection of anomalies on SCADA systems. This showed the fulfilment of the aim of the study. The objectives of this project were met, and the result showed that model outperformed other models in ways such as; this model is distributed, it can detect unknown anomalies, any bird in the flock can detect, there is no leader and many more.

REFERENCES

- Abadi, M and Jalali, S. (2006) 'An Ant Colony Optimization Algorithm for Network Vulnerability Analysis'. *Iranian Journal for Electrical and Electronics Engineering*.
- Ackley D, (2014) Artificial Life for Bigger and Safer Computing: *The John von Neumann public Lecture Series in Complexity and Computation*.
- Ackley D, (2014) *Artificial Life for Bigger and Safer Computing*: The John von Neumann public Lecture Series in Complexity and Computation.
- Adkins, F. et al (2013) *Heuristics Malware Detection via Basic Block Comparison*
- Adobe, (2010) *Adobe investigated Corporate Network Security Issue*, [online]. Available at: http://blogs.adobe.com/conversations/2010/01/adobe_investigates_corporate_n.html. Last accessed 2.12.2017.
- Agarwal T, (2014) *Know all about SCADA System Architecture and Types with Applications*. Available at: http://www.edgefxkits.com/blog/scada-system-architecture-types-applications/. Accessed 23.12.2014.
- Aljarah, I and Ludwig, S (2013) *MapReduce Intrusion Detection System based on a Particle Swarm Optimisation Clustering Algorithm*, In proceeding of 2013 IEEE Congress on Evolutionary Computation, P. 955-962, Cacun Mexico.
- Almaatouq, A., Alabdulkareem, A., Nouh, M., Alsaleh, M., Alarifi, A., Sanchez, A., Alfaris, A., Williams, J., (2014) *A Malicious Activity Detection System Utilising Predictive Modelling in Complex Environment*. 2014 IEEE 11th Consumer Communications and Networking Conference (CCNC)
- Ambler, S. (2017) *An Introduction to Agile Modelling* [online] Available from: www.agilemodeling.com/essays/introductionToAM.htm last accessed 25.08.2017
- Amudha, P. Karthik,S. and Sivakumari (2015) A Hybrid Swarm Intelligence Algorithm for Intrusion Detection Using Significant Features, *Scientific World Journal*, vol. 2015, PP.1-16.
- Anderson, E. (2003) *The Brain Behind the Enigma Code Breaking Before the Second World War*, Karlskrona, Sweden.
- Anderson, J. (1980) *Computer Security Threat Monitoring and Surveillance*, Washington, USA.

- Anderson, P., (1980) *Computer Security Threat Monitoring and Surveillance*, Technical Report, Anderson Co,
- Anjum, F., Subhadrabandhu, D., and Sarkar, S., (2003) *Signature Based Intrusion for Wireless Ad Hoc Network*: A Comparative Study of Various Routing Protocols. Proceedings of IEEE Technology Conference.
- Antoniou, P. Pitsillides, A. Blackwell, T and Engelbrecht, A (2009) *Employing the Flocking Behavior of Birds for Controlling Congestion in Autonomous Decentralized Network.*
- Apache Hadoop (2014) *Hadoop* [online], Available from: https://hadoop.apache.org. accessed 08.12.2017.
- Apache, (2017) *Apache Flume 1.8.o User Guide* [online] Available from: http://flume.apache.org/FlumeUserGuide.html. Last accessed 27.02.2018
- Aulbach S. et al (2008 p.1195-1206) *Multi-tenant databases for software as a Se*rvice, in proceedings of the ACM SIGMOD International Conference on Management of Data
- Automation World (2016). *Industrial Network Market Share 2916 According to HMS* [online] Available from: https://www.automationworld.com/article/industrial-network-market-shares-2016-according-hms. Last accessed 07.12.2017
- Babovic E, and Velagic J, (2009) *Lowering SCADA development and implementation costs using PtP concept*. Information, Communication and Automation Technologies, 2009. ICAT 2009. XXII International Symposium on 29-31 oct.2009. P.1-7, Bosnia.
- Ball, T., (2017) *Top 5 Critical Infrastructure Cyber Attack* [online] Available from: https://www.cbronline.com/cybersecurity/top-5-infrastructure-hacks/. Last accessed 13.03.2018
- Ballerini M, et al. (2008) *Interaction ruling animal collective behavior depends on topological rather than metric distance:* evidence from a field study. Proc. Natl Acad. Sci. USA **105**, P. 1232–1237
- Ballerini, M. Calbibbo, N., Candeleir, R., Cavagna, A., Cisbani, E., Giardina, I., Lecomte, V., Orlandi, A., Parisi, G., Procaccini, A., Viale, M., and Zdravkovic, V. (2008) Interaction ruling animal collective behavior depends on topological rather than metric distance: Evidence from a field study. *Proceedings of the National Academy of Sciences of the United States of America* 105: 1232–1237.

- Ballerini, M., Calbibbo, N. R. Candeleir, Cavagna, A., Cisbani, E., Giardina, I., Lecomte, V,. Orlandi, A., Parisi, G., Procaccini, A., Viale, M. and Zdravkovic V. (2008) *Interaction ruling animal collective behaviour depends on topological rather than metric distance*: Evidence from a field study. *Proceedings of the National Academy of Sciences of the United States of America* 105: pp.1232–1237
- Bashir, R., Lee, S., Khan, S., Chang, V., and Farid, S. (2016) UML Model Consistency Management: Guidelines for Software Quality Manager. *International Journal of Information Management*, 36, P.883-899.
- Baynes, C., (2018) *Austria Train Crash: At Least one Dead and More than 22 People Injured in Niklasdorf Rail Collision*. Independent [online] Available from: http://www.independent.co.uk/news/world/europe/austria-train-crash-latest-updates-dead-injured-niklasdorf-rail-police-emergency-a8206706.html. Last accessed 16.02.2018
- Belesford, D. (2011) *Exploiting Siemens Simatic S7 PLCs:* Black Hat USA+2011, NSS Labs
- Beni, G., and Wang, J., (1989) Swarm Intelligence in Cellular Robotic Systems in Proceedings of NATO Advanced Workshop on Robots and Biological Systems, Italy.
- Bere, H., and Muyingi, H. (2015) *Initial Investigation of Industrial Control System (ICS) Security Using Artificial Immune System* (AIS). IEEE, P.79-84
- Bi, J., Zhang, K., and Cheng, X., (2009) *Intrusion Detection Based on RBF Neural Network,* IEEE International Symposium on Information Engineering and Electronic Commerce.
- Bialek, W. et al (2012) *Statistical Mechanics for natural Flocks of birds*. *Proceedings of the National Academy of Sciences*. 109 (13) p. 4786- 4791.
- Blanislav A, et al (2011) *High-Performance Networked SCADA Architecture for Safety-Critical Systems*. Second Eastern European Regional Conference on the Engineering of Computer Based System.
- Blum, C., and Li, X., (2008) *Swarm intelligence in optimization, in: C. Blum, D. Merkle* (Eds.), Swarm Intelligence: Introduction and Applications, Springer Verlag, Berlin, P. 43–86.
- Boese, K., and Kahng, A., (1993) *Simulated Annealing of Neural Networks*: The Cooling Strategy Reconsidered. IEEE International Symposium on Circuits and Systems, Chicago, USA.

- Bonabeau, E. et al (1999) *Swarm Intelligence:* From Natural to Artificial System. Oxford University Press, Oxford, UK.
- Bonabeau, E. et al (1999) *Swarm Intelligence: From Natural to Artificial System*. Oxford University Press, Oxford, UK
- Boyer S, (2009) Scada: *Supervisory Control And Data Acquisition*, 4th Edition, International Society of Automation, USA.
- Boyer S, (2009) Scada: *Supervisory Control And Data Acquisition, 4th Edition, International Society of Automation*, USA.
- Bradley, T. (2016) *Zero Day Exploits. Holy Grail of the Malicious hacker* [online] Available from; https://www.lifewire.com/zero-day-exploits-2487435 last accessed 05.12.2017
- Brox, A., (2002) *Signature Based or Anomaly Based Intrusion Detection*: The Practice and Pitfalls [online] Available from. https://www.scmagazine.com/signature-based-or-anomaly-based-intrusion-detection-the-practice-and-pitfalls/article/548733/. Last accessed 18.12.2017
- Bruneau, G. (2001) *SANS Institute INFOSec Reading Room: The History and Evolution of Intrusion Detection.*
- Cambridge Advance Learners Dictionary, (2017) *Malicious* [online] Available from: https://dictionary.cambridge.org/dictionary/english/malicious. Last accessed 02.12.2017
- Camperi, M. Cavagna, A. Giardina, I. Parisi, G. Silvestri, E (2012) *Spacially Balanced Topological Interaction Grants Optimal Cohesion in Flocking Model*. *Interface Focus* 2 926) pp. 715-725
- Carafano, J. (2012) *Wiki at War: Conflict in a Socially Networked World*. Texas A & M University Press, College Station, USA.
- Carlsson, T. (2017) *Industrial Ethernet and Wireless are Growing fast-Industrial Network Market Share 2017* According to HMS [online] Available from: https://www.anybus.com/about-us/news/2017/02/20/industrial-ethernet-and-wireless-are-growing-fast-industrial-network-market-shares-2017-according-to-hms. Last accessed 07.12.2017
- Cavagna, A. (2015) *Public Lecture 2- Andrea Cavagna*: The Seventh Starling: The Wonders of Collective Animals.

- Cavagna, A. Cimarelli, A. Giardina, I, Parisi, G. Santagati, R. Stefanini, F and Viale, M. (2009) *Scale-free correlations in birds flocks*
- Cavagna, A. Cimarelli, A. Giardina, I, Parisi, G. Santagati, R. Stefanini, F and Tavarone, R. (2010) *From Emperical Data to Inter-Individual Interactions*: Unveiling the Rules of Collective Animal Behavior: Mathematical Models and Methods in Applied Science, Vol.20, PP.1491-1510.
- Cavagna, A., Cimarelli, A., Giardina, I., Parisi, G., Santagati, R., Stefanini, F., and Tavarone, R., (2010) "From Emperical Data to Inter-Individual Interactions": Unveiling the Rules of Collective Animal Behavior: *Mathematical Models and Methods in Applied Science*, Vol.20, P.1491-1510.
- Cena, G., Bertolotti, I., Hu, T. & Valenzano, A. (2014), *Design, verification, and performance of a Modbus-can adaptation layer*, in '10th IEEE Workshop on Factory Communication Systems', Toulouse
- Chandola, V. Banerjee, A. and Kumar, V. (2007) *Anomaly Detection*: A Survey, [online]. Available from: http://www.cs.umn.edu/tech_reports_upload/tr2007/07-017.pdf. Accessed 15.1.2017.
- Chen Y, Abraham A, and Yang B (2007) Hybrid flexible neural-tree-based intrusion detection systems. *International Journal of Intelligent Systems*, 22:P.337–352,
- Chen Y, and Robert J, (2005) *The Evolution of Viruses and Worms*. Available at: http://vxheaven.org/lib/pdf/The%20Evolution%20of%20Viruses%20and%20Worms.pdf. Accessed 11.12.2017.
- Ching, H., and Huang., S. (2010) *Neural Networks Based Detection of Stepping Stone Intrusion*. Expert Systems with Applications. Vol.37, P.1-35.
- Christensen, S. Jorgensen, J. Kristensen, L. (1997) *Design/CPN A Computer Tool for Coloured Petri Nets*. Brinksma, E. Editor, Tool and Algorithms for the Construction and Analysis of Systems. Vol. 1217, pp. 209-233. Springer Verla.
- Christodorescu M, et al (2008) *Mining specifications of malicious behavior*, in Proceedings of the 1st India software engineering conference. ACM, New York, USA;
- Chung, E. (2014) The Role of Espionage in Foreign *Relation*. Harvard Model Congress.
- Clerc, M. (1999) *The swarm and The queen: Towards a Deterministic and Adaptive particle Swarm Optimization*. *Proceedings of the Congress on Evolutionary Computation.* Piscataway, NJ: IEEE service center.

- Clerc, M.a nd Kennedy, J. (2002) *The particle Swarm-Explosion, stability, and Convergence in a Multidimensional Complex Space. IEEE Trans. Evolutional Computation.*
- Cloudera (2016) *Cloudera Manager API*: Installation: Basic Usage [online] Available from: http://cloudera.github.io/cm_api/docs/python-client/. Last accessed 02/01/2017
- Coates G, et al (2010) *A Trust System Architecture for SCADA Network Security.* IEEE Transaction on Power Delivery, 25 (1) P 158-169.
- Coats, D., (2017) *World Wide Assessment of the US Intelligence Community*. Senate Selected Committee of Intelligence.
- Cole, M., (2010) *Belgian Train Crash: Eighteen People Dead in Halle*. BBC News Channel.
- Control Engineering (2011*) Industrial Ethernet Protocols international market ranking* [online] Available from: http://www.controleng.com/single-article/industrial-ethernet-protocols-international-market-rankings/ef6b77489713b11cb0980f257a0c506b.html. Last accessed 16.01.2017
- Couzin, I, Krause, J. James, R, Ruxton, G. and Franks, N. (2002) *Collective Memory and Spatial Sorting in Animal Froups*
- Couzin, I. (2009) *Collective cognition in animal groups*. Trends in Cognitive Sciences
- CPN Tool (2015) *Tool for Editing Analysing Coloured Petri Nets* [online] Available from: http://cpntools.org/. Last accessed 25.08.2017
- Cruz, K. (2015) *6 Reasons Why Hackers Want to Hack Your Website*: What Do These Hackers Want from Me? [online] Available from: https://www.cloudbric.com/blog/2015/10/6-reasons-why-hackers-want-to-hack-your-website/. Last accessed 05.12.2017
- Csany, D., (2013) *3 Generations of SCADA System Architectures You Should know about.* Industrial Automation [online] Available from: http://electrical-engineering-portal.com/three-generations-of-scada-system-architectures. Last accessed 17.02.2018
- Daniel J, et al (2014) *Implementation of a novel SCADA architecture for a 210 MW Thermal Power Plant*: International Conference on Data Science & Engineering. Thiruvananthapuram, India.
- Danos, V. (2007) *Agile Modelling of Cellular Signalling*. In proceeding of AIP Conference. Vol. 963 (2) pp. 611-614

- Daoud, E (2013) 'Intrusion Detection Using a New particle Swarm method and Support Vector Machines'. *International Scholarly and Scientific Research & Innovation* 7(5) p. 55-62
- Davies, J and Arkin, R (2012) *Mobbing Behavior and Deceit and its role in Bio-inspired Autonomous Robotic Agents*. Technical Report GIT-MRL-12-02.
- Davies, K., (2015) *Why We Are Losing to Zero Day Cyber Attack* [online] Available from: http://www.fico.com/en/blogs/fraud-security/losing-zero-day-cyber-attacks/. Last accessed 14.03.2018
- Dehn, M. (1990) Vigilance for Predators: detection and Dilution Effects. *Behavioural Ecology and Sociobiology*, p. 337-342
- Dehn, M. (1990) Vigilance for Predators: detection and Dilution Effects. *Behavioural Ecology and Sociobiology*, p. 337-342
- Dell, (2015) *Dell Security Annual Report* [online] Available at: https://software.dell.com/docs/2015-dell-security-annual-threat-report-white-paper-15657.pdf. Last accessed 13.06.2015.
- Demsar, J and Bajec, I (2014) *Simulated Predator attack on Flocks: A Comparison of Tactics, Artificial Life, 20,* p. 343-359.
- Deng, L., Peng, Y., and Liu, C., (2017) Intrusion Detection Method Based on Support Vector machine Access of Modbus TCP Protocol. *Internet of Things (iThings) and IEEE Green Computing and Communications (GreenCom) IEEE, Cyber, Physical Computing (CPSCom) and IEEE Smart Data (SmartData)*, P.380-383.
- Denning, D. (1987) 'An Intrusion Detection Model'. *IEEE transaction on Software Engineering*, vol.13 (2) P. 222-232.
- Denning, D., (1987) *An Intrusion Detection Model*, IEEE Transactions on Software Engineering, Vol.13, P.222-232.
- Department of Justice (2016) *Manhattan U.S. Attorney Announces Charges Against Seven Iranians For Conducting Coordinated Campaign of Cyber Attacks Against U.S.* Financial Sector on Behalf of Islamic Revolutionary Guard Corps-Sponsored Entities [online] Available from: https://www.justice.gov/usao-sdny/pr/manhattan-us-attorney-announces-charges-against-seven-iranians-conducting-coordinated. Last accessed 16.02.2018
- Derigo, M and Stutzle, T (2004) *Ant Colony Optimization*. The MIT Press.

- Derigo, M. Maniezzo, V and Colorni, A (1991) *The Ant System*: Optimization by a colony of cooperating agents. IEEE Transactions on Systems, Man and Cybernetics-part B, 26 (1) PP. 29-41
- Devereux, C. Whittingham, M. Fernandez-Juricic, E. and Vickery, J (2005) Predator Detection and Avoidance by Starlings Under Differing Scenarios of Predation Risk: *Behaviour Ecology*, PP. 303-309.
- Diamond, S. and Lazarus, J. (1974) *The Problem of Vigilance in Animal Life. Brain, Behave*, Evol. p.60-79
- Dimeas, A. and Hatziargyriou, N. (2005) *A MAS architecture for microgrid control*, in proceeding to Power System.
- Dressler M, (1990) *SELECTION CRITERIA FOR TRANSGAS SCADA HOST ARCHITECTURE*, Saskatchewan, Canada.
- East, S., Butts, J., Papa, M., and Shenoi, A., (2009) *A taxonomy of attacks on the DNP3 protocol*, [in Critical Infrastructure Protection III, vol. 311, C. Palmer and S. Shenoi, Eds. Boston, MA: Springer-Verlag, pp. 67–81.
- Eberhart, R. and Kennedy, J. (1995 pp. 39-43) *A new optimizer using particles swarm theory. Proceedings of the sixth International Symposium on Micro Machine and Human Science.* Nagoya, Japan,
- Ebert, A., Wu, P., Mengersen, K., and Ruggeri, F. (2017) *Computationally Efficient Simulation of Queues*: The R Package queuecomputer. arXiv: 1703.02151.
- Eckstein, C. (2014) *Router Backdoors – Can you Trust your Vendor?* SANS Institute, InfoSec Reading Room.
- Ehret, C., and Ultes-Nitsche, U., (2009) *Immune System Based Intrusion Detection System*, Technical Report, http://icsa.cs.up.ac.za/issa/2008/Proceedings/Full/50.pdf last accessed 16.12.2017
- Elngar, A. Mohamed, D. Ghaleb, F (2013) 'A Real-time anomaly Network Intrusion Detection With High accuracy' *Inf Sci Lett Int J* 2 (2) p. 49-56
- Dickinson, D., (2013) Protecting Water Industry Control and SCADA Systems from Cyber Attacks [online] Available from https://www.automation.com/pdf_articles/WaterIndustryCyberSecurity_final.pdf. Last accessed 20.03.2018
- Eslamnezhad, M., and Varjani, A., (2014) *Intrusion Detection Based on MinMax K-means Clustering*. IEEE, 2014

- Farmer, D., Packard, N., and Perelson, A., (1986) *The Immune System. Adaptation and Machine Learning.* Physica, 22D. P.187-204. North Holland Amsterdam.
- Farrugia, P. et al (2010). Practical tips for surgical research: Research questions, hypotheses and objectives Can J Surg 53 (4), P. 278– 281, Medline
- Fenet, S. and Hassas, S (2001) *A Distributed Intrusion Detection and Response System Based on Mobile Autonomous Agents Using Social Insect Communication Paradigm*. In Proceedings of the first International workshop on Security of Mobile Multi Agent System (SEMAS) p.41-58
- Fenet, S. and Hassas, S. (2002) *A distributed Intrusion Detection and Response System based on mobile autonomous agents using social insects communication paradigm*. Electronic Notes in Theoretical Computer Science, p. 41-58.
- Fernandez-Juricic E. Siller S. and Kacelnik A. (2004) *Flock Density, Social Foraging and Scanning*: An Experiment With Starlings. Behav Ecol 15, PP. 371-379.
- Ferris, T. (2009) *On the Methods of Research for System Engineering*: 7th Annual Conference on Systems Engineering Research, CSER
- Forgy, L., (1965) *Cluster Analysis of Multivariant Data: Efficiency Versus Interpretability of Classifications*. Biometrics, vol. 21, P.768-769.
- Fovino, I et al (2009) *A Secure and Survivable Architecture for SCADA Systems*. Proceedings of 2009 Second International Conference on Dependability.
- Frei S, (2013) *Vulnerability Threat Trends*: A decade in Review, Transition on the way. Available at: https://www.nsslabs.com/sites/default/files/publicreport/files/Vulnerability%20Threat%20Trends.pdf. Last accessed 2.12.2017.
- Fruhlinger, J., (2013) *What is Ransomware? How it Works and How to Remove it* [online] Available from: https://www.csoonline.com/article/3236183/ransomware/what-is-ransomware-how-it-works-and-how-to-remove-it.html. Last accessed 14.03.2018
- Fusano, A. Sato, H and Namatame, A (2011) *Study of Multi-agent Based Combat Simulation for Orientation of OODA Loop.*
- Galvin, E., (2016) *Applying Threat Intelligence to Industrial Control Systems Security* [online] Available from: https://www.fireeye.com/blog/products-and-services/2016/07/applying_threat_inte.html. Last accessed 13.03.2018

- Garay, X., (2014) *The Queuing Theory Calculator* [online] Available from https://www.supositorio.com/rcalc/rcalclite.htm. Last accessed 23/01/2018
- Garnier, S. Gautrais, J. and Theraulaz, G. (2007) *Biological Principles of Swarm Intelligence*. Springer Science + Business Media.
- Gasser, L (2001) *Perspectives on organizations in multi-agent systems*. Springer-Verlag, Berlin, Germany.
- Gavazzi, C., (2010) Communication Protocol: Version 1Revision 2 [online] Available from: https://www.ccontrols.com/support/dp/CarloGavazziEM21.pdf. Last accessed 16.03.2018
- Georgeff, M. Pell, B. Pollack, M. Wooldridge, M. Tambe, M. (1999) *Belief-Desire-Intention Model of Agency, in proceedings of the 5th International Workshop on Intelligent Agent V: Agent Theories, Architectures, and Languages (ATAL-98),* p.1-10. Springer-Verlag: Heidelberg, Germany.
- Ghorbani, A et al (2010) *Network Intrusion Detection and Prevention: Concept and Techniques, Advances in Information Security*, Springer Science + Business Media.
- Ghosh A, and Schwartzbard A (2013) *A Study in using Neural Network for Anomaly and Misuse Detection.*
- Gibson, J. Rondeau, R. Eveleigh, D and Qing Tan (2012) *Benefits and Challenges of Three Cloud Computing Service Models*. In proceedings of 2012 Fourth International conference on Computational Aspects of Social Networks (CASoN), P.198-205,
- Glueck, E (1987) *An Experimental Study of Feeding, Vigilance and Predator Avoidance in a Single Bird*. Oecologia, PP. 268-272.
- Goldenberg, N. and Wool, A. (2013) *Accurate Modelling of Modbus TCP for Intrusion Detection in SCADA System*. Elsevier International Journal of Critical Infrastructure Protection. Vol.6 (2) P.63-75
- Goodin D, (2011) *Insulin pump hack delivers fatal dosage over the air* [online] Available from: https://www.theregister.co.uk/2011/10/27/fatal_insulin_pump_attack/. Last accessed 02.03.2018
- Gosine, A.,(2017) *Industrial Control Systems are Converging with Big Data* [online] Available from: https://www.controleng.com/single-article/industrial-control-systems-are-converging-with-big-data/7d1ba7994be74fd0ac7e17dcf96dd7f4.html.

Last accessed 13.03.2018

- Gragido, W. and Pirc, J. (2011) *Cybercrime and Espionage*. An Analysis of Subversive Multivector Threats, Elsevier, USA.
- Greenberg, A (2017) *How An Entire Nation Became Russia's Test lab for Cyber War* [online] Available from: https://www.wired.com/story/russian-hackers-attack-ukraine/ last accessed 05.12.2017
- Gregor, S. and Hevner, A. (2013) *Positioning and Presenting Design Science Research for Maximum Impact.* MIS Quarterly 37 (2), P.337-355
- Gunn, S., (1998) *Support Vector Machines for Classification and Regression*. Technical Report, Image Speech and Intelligent Systems Research Group, University of Southampton.
- Gupta, A., (2014) *The World's Worst Train Disasters* [online] Available from: https://www.railway-technology.com/features/featurethe-worlds-deadliest-train-accidents-4150911/. Last accessed 13. 03.2018
- Gupta, A., Pandey, O.J., Shukla, M., Dadhich, A., Ingle, A. and Ambhore, V., (2014). Intelligent Perpetual Echo Attack Detection on User Datagram Protocol Port 7 Using Ant Colony Optimization. In *Electronic Systems, Signal Processing and Computing Technologies (ICESC), 2014 International Conference on* (pp. 419-424). IEEE.
- Gupta, J., and Boral, S., (2010) *63 Dead, 150 Injured as Trains Collide in West Bengal.* The Time of India.
- Hagan, B., (2017) *Deliver Real Time Toll and Traffic Analytic* [online] Available from: https://hortonworks.com/blog/deliver-realtime-toll-and-traffic-analytics/. Last accessed 22.02.2018
- Halliday, J. (2012) *UK Riots Made Worse By Rolling News BBM*, Twitter and Facebook. Social Networking [online] Available from: https://www.theguardian.com/media/2012/mar/28/uk-riots-twitter-facebook. Last accessed 05.12.2017
- Hamilton, W (1971) *Geometry for the Selfish Herd*. *Journal of Theoretical Biology*, p. 295-311
- Hamilton, W. (1971) Geometry for the selfish herd: *Journal of Theoretical Biology*, Vol 31(2) PP.295-31.

- Happybase (2012) User Guide: *Establishing a Connection* [online] Available from: https://happybase.readthedocs.io/en/latest/ last accessed 23/12/2016
- Hayden, E. Assante, M. and Conway, T. (2014), An Abbreviated History of Automation & Industrial Control System and Cybersecurity. A SANS Analysist white paper.
- Hayes, M., and Capretz, M. (2015) Contextual Anomaly Detection Framework for Big Sensor Data: *Journal of Big Data*, vol.2 (2), P.56-64
- Hevner, A. et al (2004). "*Design science in information systems research*". *MIS quarterly*. Volume 28(1) P.75–105
- HM Government (2011) *Government Cloud Strategy*. [online] Available from: https://www.gov.uk/government/uploads/system/uploads/attachment_data/file/266214/government-cloud-strategy_0.pdf. Last accessed 05.12.2017
- Hortonworks, (2017) *Realtime Event Processing in Hadoop With Nifi, Kafka and Storm* [online] Available from: https://hortonworks.com/tutorial/realtime-event-processing-in-hadoop-with-nifi-kafka-and-storm/. Last accessed 22.02.2018
- http://www.theregister.co.uk/2011/10/27/fatal_insulin_pump_attack/. Accessed 10.12.2014.
- Hulley, S. et al (2007). *Designing clinical research*. 3rd ed. Philadelphia (PA): Lippincott Williams and Wilkins
- IBM. (2012) *What is big data?* [Online]. Available at: http://www-01.ibm.com/software/data/bigdata/what-is-big-data.html [Accessed: 21 August 2017]
- ICM, IPI, (2016) The Impact of New Technologies on Peace, Security and Development. *Independent Commission on Multilateralism* [online] Available from: https://www.icm2016.org/IMG/pdf/new_tech_paper.pdf. Last accessed 14.03.2018
- ICS-CERT (2016) *Year in Review. Industrial Control Systems Cyber Emergency Response Team*. NCCIC.
- ICS-CERT MONITOR (2014) *Incident response Activity*: Internet Accessible Control Systems At Risk, Available at: https://ics-cert.us-cert.gov/sites/default/files/Monitors/ICS-CERT_Monitor_%20Jan-April2014.pdf. Accessed 2.12.2017.
- ICS-CERT, (2015) *Incident Response/Vulnerability Coordination in 2014*.

- Ignat N, (2014) *Dependability and Vulnerability of SCADA Systems*. Bucharest, Hungary.
- Internet Security System, (2006) *SCADA Security and Terrorism: We're not Crying Wolf* [online] Available from: https://www.blackhat.com/presentations/bh-federal-06/BH-Fed-06-Maynor-Graham-up.pdf. Last accessed 14.03.2018
- Isikoff, M., and Corn, D., (2017) Russian Roulette: The Inside Story of Putin's War on America and the Election of Donald Trump [online] Available from: https://www.nytimes.com/2018/03/14/books/review/russian-roulette-michael-isikoff-david-corn.html?rref=collection%2Fnewseventcollection%2Frussian-election-hacking&action=click&contentCollection=politics&region=stream&module=stream_unit&version=latest&contentPlacement=2&pgtype=collection. Last accessed 15.03.2018.
- Jang, J and Brumley, D. (2009) "*BitShred: Fast, Scalable Code Reuse Detection in Binary Code*". CMU-CyLab.
- Jensen, K. (1987) *Coloured Petri nets*. Petri nets: Central Model and Their Properties. pp.248-299, Springer, Berlin
- Jensen, K. (1997) *Coloured Petri Nets: Basic Concepts, Analysis Methods and Practical Use*, Vol.1, Monographs in Theoretical Computer Science. Springer-Verlag
- Jeong, H. Hyun, W. Lim, J. You, I (2012) *Anomaly Teletraffic Intrusion Detection Systems on Hadoop Based platforms*: A Survey of some problems and solutions in Network Based information System (NBIS) in Proceedings of 15th IEEE International conference, Melbourne, Australia p. 766-770
- Jerne, N. (1973) The Immune System. *Sci. vol*. 229, P.52-60.
- Jerne, N. (1974) *Towards a Network Theory of the Immune System*. vol.125, P.373-389.
- Jovic,M. and Hauswirth, M, (2010) *Performance Testing of GUI Applications*. In Testbeds. Second International Workshop on Testing Techniques and Experimentation Benchmarks for Event Driven Software, Paris, France.
- Joyce G, (2009) Evolution in an RNA world. *Cold spring Harbor Symposium on Quantitative Biology; 74, P.17-23.*
- Joyce G, (2009) *Evolution in an RNA world*. *Cold spring Harbor Symposium on Quantitative Biology; 74, P.17-23.*

- Karaboga, D (2005) *An Idea Based on Honey Bee Swarm for Numerical Optimization*, Technical report-TR06, Erciyes University, Faculty of Computing and Engineering.
- Karnouskos, S., and Colombo, A., (2011) Architecting the Next Generation of Service-based SCADA/DCS System of Systems in Proceedings of *37th Annual Conference of the IEEE Industrial Electronics Society (IECON 2011) Melbourne*, Australia.
- Kaspersky (2017) *History of Malicious Programs* [online] available from: https://securelist.com/threats/history-of-malicious-programs/ Last accessed 02.12.2017
- Kaspersky (2017) *The Sate of Industrial Cybersecurity 2017* [online] Available from: https://go.kaspersky.com/rs/802-IJN-240/images/ICS%20WHITE%20PAPER.pdf. Last accessed 12.12.2017
- Kattas, G. Xu, X. and Small, M. (2012) *Dynamic Modelling of Collective Behaviours From Pigeon Flight Data*: Flocks Cohesion and Dispersion. *Journal of Computational Biology*.
- Kenneally, J., Mulqueen, K., and Wai, J. (2014) Government Cloud Computing: Planning for Innovation and Value. Intel [online] Available from: https://www.intel.com/content/dam/www/public/us/en/documents/white-papers/cloud-computing-government-paper.pdf. Last accessed 15.03.2018
- Khan, A., (2015) *Consistency of UML Class, Object, and Statechart Diagrams Using Ontology Reasoners*, J. Vis. Lang. Comput. P.42-65
- Kiss, I., Genge, B., Haller, P., and Sebestyen, G., (2014*) Data Clustering Based Anomaly Detection in Industrial Control Systems*. In proceedings of Intelligent Computer Communications and Processing (ICCP). IEEE, P.275-281,
- Kleinmann and Wood (2014) *Accurate Modelling of Siemens S7 SCADA Protocol for Intrusion Detection and Digital Forensics*. JDFSL. Vol.9 (2), P. 37-50.
- Kohavi, R., and Provost, F., (1998) *Machine Learning*. Vol.30 (2), P.271-274
- Kontarinis, D., (2016) *Debate in a Multi-Agent System*: Multiparty Argumentation Protocols. Multi-agent System (cs.MA).
- Kothari, C. (2004) Research Methodology: Methods and Techniques, 2nd ED, New *Age International Publisher*, Jaipur, India.
- Krishna, R. (2014) *Hadoop Course* [online], Available from: http://www.durgasoft.com/Hadoop%20ramakrishna%20madhapur.asp. Last accessed 17.01.2017.

- Kristensen, L. Kristensen, S and Jensen, K. (1998) The Practitioner's guide to Coloured Petri Nets. *International Journal on Software Tools for Technology Transfer,* Vol.2 pp. 98-132.
- Kubarkov, Y., and Makarov, Y., (2016) Application of Multi Agent in Solving Problems for Operation of a Smart Grid. International conference on *Industrial Engineering, Applications and Manufacturing (ICIEAM),* Russia
- Kummer, O. (2002) *Reference Nets*. Logos Verlag
- Kushner D, (2013) *The real Story of Stuxnet*: How Kaspersky Lab tracked down the malware that stymied Iran's nuclear-fuel enrichment program, [online]. Available at: http://spectrum.ieee.org/telecom/security/the-real-story-of-stuxnet. Last accessed 2.12.2017.
- Lam, C. (2011) *Hadoop in Action*. Manning publishing, Stanford, United State.
- Langner R, (2011) *Stuxnet*: Dissecting Cyberwarfare Weapon. Available at: http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=5772960. Last accessed 07.12.2017.
- Langner, R. (2013) *To Kill a Centrifuge*. A Technical Analysis of What Stuxnet Creators Tried to Achieve.
- Langton, C. (1986)."*Studying artificial life with cellular automata*." Physica D: Nonlinear Phenomena 22(1-3): 120-149.
- Langton, C. (2000) *Artificial Life*: An Overview, MIT Press, Massachusetts.
- Lavenberg, S. (1983) *Computer Performance Modelling Handbook*. Academic Press, New York, USA.
- Lawlor, E. (1993) *Discover Nature Close to Home*: Things to Know and Things to do. STACKPOLE BOOKS, Harrisburg.
- Lenzen, C and Radeva, T (2013) *The Power of Pheromones in Ant Foraging*
- Levenson, E., Arif, M., and Elsayed, H., (2017) *Train Collision in Egypt Leaves Dozens Dead*. CNN.
- Lianying, Z and Fengyu, L (2006) 'A swarm intelligence based Intrision Detection Technique'. *IJCSNS international Journal of Computer Science and Network Security*, 6 (7) p. 146-150.
- Liker, J. (2004) The Toyota Way: *The Right Process Will Produce The Right Results*, United State. McGraw-Hill.

- Lima, S. and Dill, M (1990) *Behavioural Decisions made under the risk of prediction*: a review and prospectus. *Can.j.zool,* 68, p.619-640.
- Linda O., Vollmer T., and Manic, M., (2009) *Neural Network Based Intrusion Detection System for Critical Infrastructures*. In Proceedings of the 2009 international joint conference on Neural Networks, IEEE Press, Piscataway, NJ, USA, P.102–109.
- Lindstrom, B. and Wells, L. (1999) *Performance Analysis Using Coloured Petri Nets*. Master Thesis, University of Aarhus.
- Lomazova, I and Schnoebelen, P. (2000) *Some Decidability Results for Nested Petri nets,* pp.208-220, Springer LNCS 1755
- Lu, X., Wang, W., and Ma, J., (2013) *An empirical study of communication infrastructures towards the smart grid: Design, implementation, evaluation*, IEEE Trans. Smart Grid, vol. 4 (1), P. 170 -183.
- Lynch, S. and Rajendran, K. (2004) *Design Diagrams for Multi-Agent Systems:* 16th Workshop of the psychology of programming Interest Group. Carlow, Irland,
- M. Ballerini, M., Calbibbo, N., Candeleir, R., Cavagna, A., Cisbani, E., Giardina, I., Lecomte, V., Orlandi, A., Parisi, G., Procaccini, A., Viale, M., and Zdravkovic. V., (2008) "Interaction ruling animal collective behavior depends on topological rather than metric distance: Evidence from a field study". *Proceedings of the National Academy of Sciences of the United States of America* 105, P.1232–1237.
- Mafra, Moll, and Fraga (2010) *Octopus IIDS: An Anomaly Based Intelligent Intrusion Detection System*, IEEE
- Maisel W, and Kohno T, (2010) *Improving the security and privacy of implantable*
- Malaska, T.(2015) *Architectural Patterns for Near real-Time Data Processing with Apache Hadoop*
- March, S. and Smith, G. (1995) *Design and natural Science Research on Information Technology*. Decision Support Systems, 15 (4) P.251-266
- Martin, G. (1986) The Eye of a Passeriformes Bird, The European Starling (*Sturnus Vulgaris*): Eye Movement Amplitude, Visual Fields and Schematic Optics: Journal of Comparative Physiology A, !59, PP.545-557
- Martin, G., (1986). *The Eye of a Passeriform Bird, The European Starling Eyes Movement Amplitude*, Virtual Fields and Schematics, Optics, J Comp Physiol A, P.545-557.

- Marzolla, M., Mamprin, R., and Balsamo, S. (2004) *Performance Evaluation of Software Architecture with Queuing Network Models.* In Proceedings of European Simulation and Modelling Conference, In Carmen Bobenau, (ESMc), P.206-213, UNESCO, Paris, France.
- Matrosov A, et al (2011) *Stuxnet Under The Microscope* , Revision 1.31.
- McClanahan R, (2002) *The Benefit of Networked SCADA Systems Utilizing IP-Enabled Networks*. Arkansas, USA.
- McClanahan R, (2002) *The Benefit of Networked SCADA Systems Utilizing IP-Enabled Networks*. Arkansas, USA
- McCormick, M. (2012) *waterfall Vs. Agile Methodology* [online] Available from: www.mccormickpcs.com/images/Waterfall_vs_Agile_Methodology.pdf last accessed 25.08.2017
- McMillen, D. (2016) *Attacks Targeting Industrial Control Systems* (ICS) UP 110 Percent. Security Intelligent [online] Available at: https://securityintelligence.com/attacks-targeting-industrial-control-systems-ics-up-110-percent/. Last accessed 12/02/2018
- Mehta, N., and Sahai, A., (2017) *Internet of Things: raging Devices and Standardizations in Low-powered Protocols*. Second International Conference on Electrical, Computer and Communication Technologies (ICECCT), P.1-5.
- Mell P and Grance T (2009) *The NIST definition of Cloud Computing*,[internet] Available from: http://csrc.nist.gov/groups/SNS/cloud-computing/. Last accessed 03.04.2017.
- Miru, G. (2017) *The Siemens S7 Communication* – Part 2 Job Requests and Ack data [online] Available from: http://gmiru.com/article/s7comm-part2/. Last accessed 07.12.2017
- Modbus (2006) Modbus Application Protocol Specification [online] Available from: http://www.modbus.org/docs/Modbus_Application_Protocol_V1_1b.pdf. Last accessed 16.03.2018
- Moreno, G. and Merson, P. (2008) *Model Driven Performance Analysis* in Proceeding of 4th International Conference on the Quality of Software Architecture. (QoSA), vol.5281, P.135-151, Germany

- Morris, T., Vaughn, R., and Dandass, Y. (2012) *A Retrofit Network Intrusion Detection System for Modbus RTU and ASCII Industrial Control Systems*. IEEE 45th Hawaii International Conference on System Sciences, P. 2338 – 2345.
- Nguyen, B. (2013) *Cloud and Internet Security*: Security Matters. Version 1.43
- Nguyen, P. Lee, S and Ngo, V (1995) *Effect of Vision Angle on the Phase Transition in a Flocking Behaviour of Animal Groups*.
- Nourian, A., and Madnick, S., (2018) *A Systems Theoretic Approach to the Security Threats in Cyber Physical Systems Applied to Stuxnet*. IEEE Transactions on Dependable and Secure Computing.
- Novak P et al, (2013) *Architecture of a Multi-Agent System for SCADA Level in Smart Distributed Environment.*
- Ober, I., Graf, S., and Ober, I., (2004) *Model Checking of UML Models via a Mapping to Communicating Extended Timed Automata*. 11th International SPIN Workshop on Model Checking of Software. Volume LNCS 2989, P.127-145
- Partridge, B and Pitcher, J (1980) *The Sensory basis for Fish School*: relative Roles of Lateral line and Vision. Journal of Comparative Physiology P. 315-325.
- Patil, M. and Yogi, N. (2011) Importance of Data Collection and validation for Systematic Software Development Process. *International Journal of Computer Science and Information Technology* (IJCSIT) Vol. 3 (2), P. 260-278.
- Patron, S., Yurcik, W. and Doss, D. (2001) *An Achilles Heel in Signature Based IDS: Squealing False Positive in SNORT*. In Proceedings of the 4th International Symposium on Recent Advances in Intrusion Detection.
- Peffers, K. et al, (2007) A Design Science Research Methodology for Information Systems Research. *Journal of Management Information System.* Volume 24 (3), P. 45-78
- Perrin, C., (2010) *The Danger of Complexity: More Code, More Bugs* [online] Available from: https://www.techrepublic.com/blog/it-security/the-danger-of-complexity-more-code-more-bugs/. Last accessed 14.03.2018
- Pfleeger, C. and Pfleeger, S. (2003) *Security in Network*: Intrusion Detection Systems [online] Available from:

http://www.informit.com/articles/article.aspx?p=31339&seqNum=5. Accessed 15.01.2015.

- Philips, L et al (2006) *Agent based control of distributed infrastructure resources*. Available at: http://www.sandia.gov/scada/documents/sand_2005_737.pdf. [online] Accesed 08.01.2017.
- Pimentel B, (2010) *Juniper Networks Investigating cyber-attacks*, [online] Available at: http://www.marketwatch.com/story/juniper-networks-investigating-cyber-attacks-2010-01-15. Accessed 2.12.2017.
- Pipattanasomporn, M et al. (2009) *Multi-Agent System in a Distributed Smart Grid*: Design and Implementation.
- PIPE (2013) *Platform Independent Petri Nets Editor* [online] Available from: http://pipe2.sourceforge.net// Last accessed 25.08.2017
- Pomeroy, H and Heppner, F (1992) *Structure of Turning in Airborne Rock Dove (Columba Livia)* Flocks, The Auk 109 (2) PP.256-267
- Positive Technologies (2014) SCADA SAFETY IN NUMBERS [online]. Available from: http://www.ptsecurity.com/download/SCADA_analytics_english.pdf. Accessed 2.12.2017.
- Potts, W. (1984) *The chorus-line hypotheses of manoeuvre coordination in a Avian flocks*: Nature 309, pp. 344-345.
- Powell, G (1974) 'Experimental analysis of the social value of flocking by starlings (Sturnus vulgaris) in relation to prediction and foraging'. *Animal Behaviour*, 22 p.501-505.
- Powell, G, (1974). *Experimental Analysis of Social value of Flocking by Starlings* (*Sturnus Vulgaris*) *in Relation to Predation and Foraging*, Amin Behav 22:501-505
- Prasad, S. Rao, A. and Rehani, E. (2001) *Developing Hypothesis and Research Questions.*
- Prigg M, (2014), *The ambulance drone that could save your life*: Flying defibrillator can reach speeds of 60mph. available from: http://www.dailymail.co.uk/sciencetech/article-2811851/The-ambulance-drone-save-life-Flying-defibrillator-reach-speeds-60mph.html. Accessed 06.12.2017.
- Pulliam, H. (1973) *On the Advantages of Flocking*. *Journal of The oretical Biology*, p. 419-422.

- Pulliam, H. (1973) *On the Advantages of Flocking*. *Journal of The oretical Biology*, p. 419-422.
- Rainie, L. and Anderson, J., (2017) *The Internet of Things Connectivity Binge*: What are the Implications? [online] Available from: http://www.pewinternet.org/2017/06/06/the-internet-of-things-connectivity-binge-what-are-the-implications/. Last accessed 13.03.2017
- Rajasekar, S. Philominathan, P. and Chinnatambi, V. (2013) *Research Methodology*: *Physics ed-Ph* 14. P.1-53
- Ratzer, A., Wells, L., Lassen, H., Laursen, M., Qvortrup, J., Stissing, M., Westergaard, M., Christensen, S., Jensen, K. (2003) *CPN Tools for Editing, Simulating, and Analysing Coloured Petri Nets*. In Proceedings of the 24th International Conference on Applications and Theory of Petri Nets (ICATPN 2003), Vol. 2679, P.450–462, Springer-Verlag, Eindhoven, Netherlands
- Raumbaugh, J. Jacobson, I and Booch, G. (2004) *Unified Modelling language* Reference manual: 2nd Edition, Pearson Higher Education.
- Raynolds, C. [2015] *Boids* [online] Available from: www.red3d.com/cwr/boids/ last accessed 19/01/2017
- Read, H. Xynos, K and Blyth, A (2009) *Presenting DEViSE: Data Exchange for Visualising Security Events*.
- Reese, G. (2009) *Cloud Application Architectures: Building Applications and Infrastructure in the Cloud*, O'Reilly Media, Sebastopol USA.
- Reynoid, C. (1987 p.25-34) *Flocks, Herds and Schools*: A distributed behavioral model in Computer Graphics.
- Robert, G. (1995) *Why Individual Vigilance Declines as Group Size Increases*: *The Association for The Study of Animal Behaviour*. 51 p.1077-1086.
- Rock, R. (2016) *Dell Annual Threat Report Reveals Cyber Criminal Using Aggressive, Shape-Shifting Tactics*; 50% Surge in Encrypted Traffic Affected Millions of Users in 2015 [online] Available from: https://www.dell.com/learn/us/en/vn/press-releases/2016-02-22-annual-threat-report-details-the-cybercrime-trends. Last accessed 12.12.2017
- Rojon, C., & Saunders, M. (2012). *Formulating a convincing rationale for a research study*. Coaching: An International Journal of Theory, Research and Practice, 5(1), 1-7

- Rosen, A. and Hedenstrom, M. (2000) *Predator Versus Prey*: On Aerial Hunting and Escape Strategies in Birds, Oxford Journals, Behavioural Ecology, Vol. 12 (2) PP. 150-156
- Rosenberg, L., (2016) Artificial Swarm Intelligence, a Human-in-the-Loop to AI. *In proceedings of the Thirtieth AAAI Conference on Artificial Intelligence* (AAAI-16)
- Rossi, M. and Sein, K. (2003) *Design research workshop*: a proactive research approach. 26th Information Systems Research Seminar in Scandinavia, Haikko Finland: The IRIS Association
- Rumehart, D., Hinton, G., and Williams, R., (1986) *Learning Representations by Back Propagation Errors*, Nature, Vol.323, P.533-536.
- SANS institute (2001) *History of Encryption* [online] Available from: https://www.sans.org/reading-room/whitepapers/vpns/history-encryption-730. Last accessed 02.12.2017
- SANS institute (2001) *Intrusion Detection Systems*: Definition, Need and Challenges. SANS Institute InfoSec Reading Room.
- Scarfone, K. and Mell, P. (2007) *Guide to Intrusion Detection and Prevention System* (IDPS): recommendation of the National Institute of Standard and Technology.
- Schalkoff, R. (1997) *Artificial Neural Networks*, McGraw-Hill, New York.
- Schmidt A, and Albayrak S (2008) *Malicious Software for Smartphones*: Available at: https://www.dai-labor.de/fileadmin/files/publications/smartphone_malware.pdf. Accessed 10.12.2017.
- Schneier, B (2012) *Liars and outliers*: *Enabling the Trust That Society Needs To Thrive*: Technological Advances, John Wiley and Sons, USA.
- Shadmehr, R. Smith, M. and Krakauer, J (2010) *Error Correction, Sensory Prediction and Adaptation in Motor Control*, *Annual Review of Neuroscience*, New York.
- Shah, B., and Trivedi, B., (2012) *Artificial Neural Network Based Intrusion Detection System*: A Survey. International Journal of Computer Application. Vol. 39 (6), P.13-18.
- Shahzad, A.; Musa, S.; Aborujilah, A.; Irfan, M.(2013) 'A Performance Approach: SCADA System Implementation within Cloud Computing Environment', *Advanced Computer Science Applications and Technologies (ACSAT), 2013 International Conference on* , vol., no., p.274,277, 23-24

- Shi, Y. and Eberhart, R. (1998) *A Modified particle Swarm Optimizer*. *IEEE Word Congress on Computational Intelligence.*
- Shi, Y. and Eberhart, R. (1998) *Parameter Selection in Particle Swarm Optimization:* In proceedings of 1998 Annual Conference on Evolutionary Programming, San Diego, USA.
- Shield, L and Twycross, A (2003) *The Difference between quantitative and qualitative research*, Paediatric Nursing, 15(9), p. 24.
- Short, L. (1961) *Interspecies Flocking of birds in Montane Forest in Oaxaca*, Mexico. Wilson Bull, p. 341-437.
- Shosha, A et al (2011) *Detecting Cyber Intrusion in SCADA Network Using Multi-Agent Collaboration.*
- Simon, A. (1996) *The Science of the Artificial*, 3rd Edition, MIT Press, Cambridge
- Simon, H. (1996) *The Sciences of the Artificial*, 3rd Edition, MIT Press, Cambridge.
- Singer, P., and Friedman, A. (2014) *Cyber Security and Cyberwar*. What Everyone Need to Know. Oxford University Press.
- Smith, S. (2016) *The Evolution of Mobile Ransomware*. Avast blog [online] Available from: https://blog.avast.com/the-evolution-of-mobile-ransomware last accessed 05.12.2017
- Smolin L, (1997) *The Life of the Cosmos*. London: Phoenix.
- Song, X., Wu, M., Jermain, C., and Ranka, S., (2007*) Conditional Anomaly Detection.* IEEE Transactions on Knowledge and Data Engineering, Vol. 19 (5), P.631-645.
- Soy, K. (1997). *The case study as a research method*. University of Texas at Austin [online] Available from: https://www.ischool.utexas.edu/~ssoy/usesusers/l391d1b.htm. Last accessed 25.05.2017
- Spafford E, (1990) *Computer viruses, a form of artificial life*? Technical Report CSD-TR-985, Software Engineering Research Center, Purdue University. Available at: http://docs.lib.purdue.edu/cgi/viewcontent.cgi?article=1836&context=cstech. Accessed 11.12.2017.

- Steinhaus, H., (1956) *Sur La Division des Corpes Materiels en Parties*. Bulletin de l'Acad´emie Polonaise des Sciences, Classe III, vol. 4 (12), P.801-804.
- Stouffer, K., Falco, J., and Scarfone, K., (2008) *Guidance to Industrial Control Systems (ICS) Security*. NIST Special Publication 800-82.
- Sun, S. and Wang, Y. (2010) Research and application of an improved support vector clustering algorithm on anomaly detection. *Journal of Software,* 5 (3)
- Sungard (2015) *Big Data Challenges and Opportunities for the Energy Industry* [online] Available at: https://www.sungard.com/~/media/fs/energy/resources/white-papers/Big-Data-Challenges-Opportunities-Energy-Industry.ashx. Last accessed 04.08.2015.
- Suthaharan, S (2013) *Big Data Classification*: *Problem and Challenges in Network Intrusion Prediction with Machine learning:* In Big Data Analytics Workshop , ACM Sigmettrics. Pittsburgh, USA.
- Sutic, D. and Atlagic, B. (2013) 'Requirement Bottlenecks in a Cloud Based SCADA System:' *Information & Communication Technology Electrics &Microelectronics (MIPRO), 2013 36th International Conference p.857-862*
- Sycara, K. (2001) *Multi-agent infrastructure, agent discovery, middle agents for web services and interoperation*. Volume 2086, 17-49, Springer.
- Tavallaee, M. Ebrahim, B. Wei, L and Ali, A. (2009) 'A Detailed Analysis of the KDD CUP 99 Data Set', *In proceedings of IEEE International Conference on Computational Intelligence for Security and Defence Application*, p. 53-58
- Teng, S et al (2010). A cooperative network intrusion detection based on fuzzy SVMs. *Journal of Networks*, 5 (4) P.475-483.
- Thul, R., Thurley, K. and Falcke, M. (2009) Towards a predictive model of Ca2+ puffs. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, Vol.*19*(3), p.037108.
- Tinbergen, N. (1951) *The Study of Instinct. Oxford:* Oxford University Press, PP. 99-102
- Tradewell, S. and Zhou, M. (2009) *A Heuristic Approach for Detection of Obfuscated Malware.*
- Tsang, C. and kwong, S. (2005) 'Multi-Agent Intrusion Setection System in Industrial Network using Ant Colony Clustering Approach and Unsupervised Feature Extraction'. *In IEEE International Conference on Industrial Technology*, ICIT, p.51-56)

- Tucker, D., (1997) *The Federal Government's War on Economic Espionage*, 18 U. PA.J.INT'L. ECON. L. P.1109-1152.
- Ujvarosi, A., (2016) Evolution of SCADA Systems. *Bulletin of the Transilvania University of Brasov*. Vol. 9 (58)
- United Nation Congress (2015) *New and Emerging Forms of Crime: Threats the World Must Reckon with* [online] Available from: http://www.un.org/en/events/crimecongress2015/pdf/Factsheet_5_Emerging_forms_of_crime_EN.pdf. Last accessed 15.03.2018

- USA National Security Agency (2010) *A Framework For Assessing and Improving the Security Posture of Industrial Control System* (ICS). System and Network Analysis Center.
- Vacca, J.(2017) *Computer and Information Security Handbook*. 3rd Edition, Morgan Kaufmann, Elsevier, United State.
- Vaishanav, V. and Kuechler, B. (2007) *Design Science Research in Information Systems* [online] Available from: http://desrist.org/desrist/content/design-science-research-in-information-systems.pdf. Last accessed 31.05.2017
- Venkataraman, S. Brumley, D. and Sen, S and Spatscheck, O. (2013) *Automatically Inferring the Evolution of Malicious Activity on the Internet*. Research Showcase: Carnegie Mellon University.
- Verwer, A. (2010) *Introduction to ProfiNet*: Overview and Applications of ProfiNet [online] Available from: http://.profibus.com/uploads/media/profinet_overview.pdf.pdf. Last accessed 06.12.2017.
- Wegener, R. (2012) *Multi-Agent Malicious Behaviour Detection: Multi Agent System.*
- Wei, Z., and Hao-yu, W., (2010) *Intrusion Detection Systems Designed Based on Back Propagation Neural Network*, IEEE.
- Wells, L. (2006) *Performance Analysis Using CPN Tools*. ValueTools, ACM , Pisa, Italy.
- Whitten, J., Bentley, L., and Dittman, K., (2001) *System analysis and Design Methods*, 5th Edition. McGraw-Hill, Irwin.
- Wilensky, (2016) *NetLogo Multi Agent Modelling Environment* [online] Available from: https://ccl.northwestern.edu/netlogo/. Last accessed 03/02/2018

- Williamson, M. (2002) *Biological Inspired Approaches to Computer Security*: Technical Reports: HP Laboratories, Bristol.
- Wireshark, (2018) *Wireshark* [online] Available from: https://www.wireshark.org/. last accessed 23.02.2018
- Wooldridge, M. (2009). *An Introduction to Multi Agent System*, Second Edition. John Wiley and Sons, West Sussex, England.
- Wooldridge, M. and Jennings, N. R. (1994) Formalizing the cooperative problem solving process. In *Proceedings of the 13th International Workshop on Distributed Artificial Intel- ligence (IWDAI-94), Lake Quinalt, WA*, p. 403-417.
- Wooldridge, M. and Jennings, N. R. (1995) 'Formalizing the cooperative problem-solving process'. In *Proceedings of the 13th International Workshop on Distributed Artificial Intel- ligence (IWDAI-94), Lake Quinalt, WA*, p. 403-417.
- Xu, Z. Hu, Q. and Zhang, C. (2013) *Why Computer Talents Become Hackers*. Communications of the ACM, Vol.56 (4), P.64-74
- Xuyana T, (2005) *Life, Artificial Life and Generalized Artificial Life*: proceedings of international Conference on Neural Network and Brain.
- Yamijala, H., (2013) 6 Reasons Why Hadoop on the Cloud Makes Sense [online] Available from: https://www.thoughtworks.com/insights/blog/6-reasons-why-hadoop-cloud-makes-sense. Last accessed 21.03.2018
- Yardley, T., (2008) SCADA: Issues, Vulnerabilities and Future Directions [online] Available from: http://www.usenix.org/system/files/login/articles/258-yardley.pdf. Last accessed 19.03.2018.
- Yang, W., and Zhao, Q. (2014) Cyber Security Issues of the Critical Components for Industrial Control System. Guidance, Navigation and Control Conference, P. 2698-2703
- Yang, Y., McLaughlin, K., Littler, T., Sezer, S., Pranggono, B., and Wang, H. (2013) *Intrusion detection system for IEC 60870-5-104 based SCADA networks*. In IEEE Power and Energy Society General Meeting (PES), pages 1–5. doi: 10.1109/PESMG.2013. 6672100.
- Yasakethu, S., and Jiang, J., (2013) *Intrusion Detection via Machine Learning for SCADA System Protection*. Proceedings pf the 1st International Symposium for ICS and SCADA Cyber Security Research. P.101-105.

- Yin, R. (2009) *Case Study Research*: Design and Methods. Applied Social Research Methods, 4th Ed, SAGE, London.
- Zheng, Y., Zhang, H., and Yu, Y., (2015) *Detecting Collective Anomalies from Multiple Spatio-Temporal Datasets Across Different Domains*, ACM
- Zoratto, F. Santucci, D and Alleva, E (2009) *Theories Commonly Adopted to Explain The Anti-Predatory Benefits of the Group Life*: The Case of Starling (Sturnus Vulgaries). Rendiconti Lincei, PP.1-14.
- Zucker, M. (2009). *How to Do Case Study Research*. School of Nursing Faculty Publications Series [Online]. Available from: http://scholarworks.umass.edu/cgi/viewcontent.cgi?article=1001&context=nursing_faculty_pubs. Last accessed 25.5.2017
- Zuech, R., Khoshgoftaar, T., and Wald, R. (2015) Intrusion Detection and Big Heterogeneous Data: a Survey. *Journal of Big Data.* P.1-41

APPENDIXES

LOGICAL DESIGN OF THE SYSTEM

The logical design of the system will provide an abstract view of the system design that will show the flow of information from one place to another. This will explain the makeup of the Artificial Life Framework (ALF) and how the data will be processed as well as the position of the algorithm in the system. Lambda architecture will be used as it is suitable for the project because it supports both real time and batch processing of data. Similar idea was demonstrated by amazon (2015) for real time processing of batch data. The Lambda architecture is mainly based on three principles namely; 1) Fault tolerance, 2) Data immutability, where data stored will not be updated or deleted, 3) Recomputation of data that have been stored in the system, due to the fact that the system allows fault tolerance and immutability. Figure 1 is the system architecture based on Lambda and below is the explanation of various components of the architecture.

LAMBDA ARCHITECTURE

The architecture below in figure 1 was designed following the Lambda architecture for the purpose of batch and near real time processing of data. The architecture came with the capability of analysing data on the fly as well as saved data in a system. This interesting paradigm (Lambda architecture) was developed by Nathan Marz (Marz and Warren, 2013), which alleviated the problem of real time data analytic. Lambda architecture is an answer to the cry of many big data analytic for real time data processing. It is a generic architecture or framework that could be fitted into many environment in solving big data as well as analytic problems. However, the architecture is made up of three layers namely: the Batch, the Speed and the Serving layer.

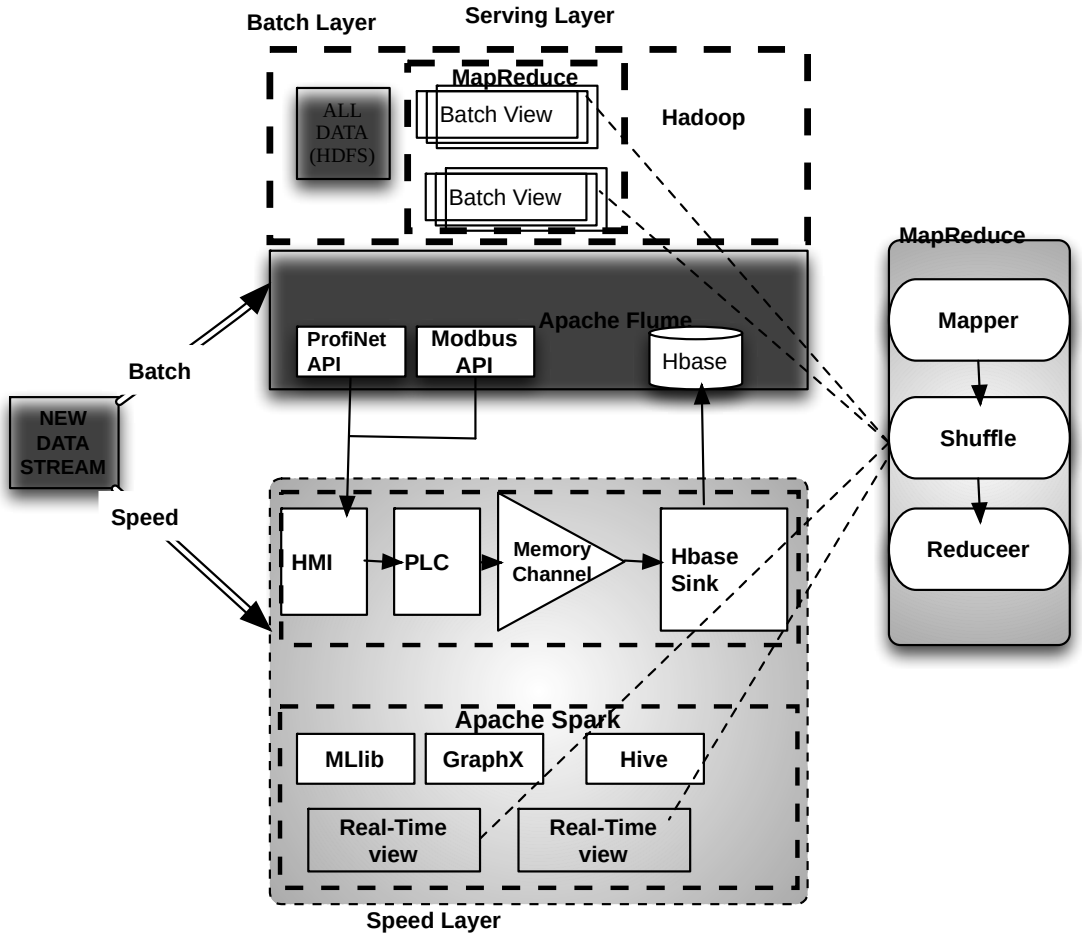


Figure 1: System Architecture Based on Lambda

Batch Layer

The Batch layer is implemented using Hadoop framework for savings and processing data in Hadoop. Hadoop is a framework from apache software foundation that was developed and named by Doug Cutting in 2006 as cited by Derrick (2009). The very fundamental architecture of Hadoop was to make distributed data storage and processing easier and cheap through the use of commodity hardware. Hadoop is an open source software implementation that has gained popularity through its distributed nature and cheap implementation. Since the architecture is designed with commodity software, failures are always expected and planed for. The two main components of Hadoop are the Hadoop Distributed File Systems (HDFS) and MapReduce. The HDFS is for data storage in Hadoop clusters through the NameNode and DataNode. When a client wants to save or store data, the first point of contact is the NameNode. The client will contact the NameNode and instructions will be issues to the client with regards to locations on the DataNode, where to store the data as seen on figure 2. The metadata will be stored on the NameNode at the same time as well as copied to the secondary NameNode as a backup. The calling to the NameNode by the client as well as the DataNode happens as if it is occurring at the

same time as seen in figure 2. The client make an RPC call to the NameNode and receives instruction as to where in the DataNode to store the data, the client will contact or stream the data straight to the DataNode.

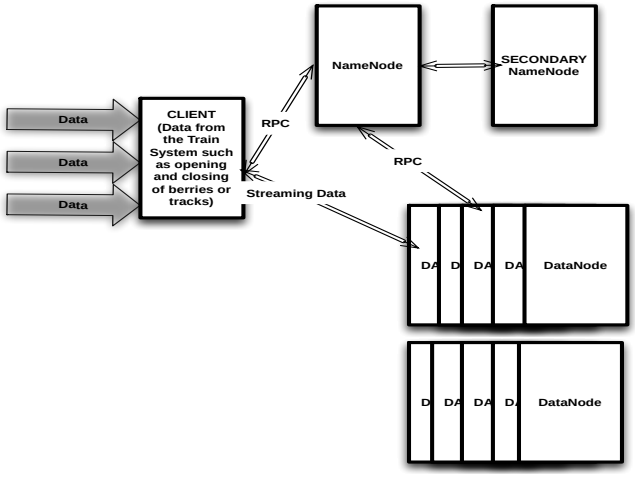


Figure 2: Batch Layer

MapReduce/Serving Layer

The serving layer or based on our architecture, MapReduce layer is the layer that is dedicated to performing the task that is stated in the algorithm. The layer has the three classes namely; Mapper, Shuffle sort or Partition and Reducer class see figure 3.

Mapper Scuffle Sort and Reducer Class

The mapper class will map out the packet by taking the packet and divide it into different sections. Using Modbus packet as an example here, the mapper class for a packet that contains 13 01 006A 0002 7687. In order to interpret the packet that contains these data;

-13 is the slave ID address responding to the master and it is in hexadecimal, which the decimal value will be 19.

-The 01 is the function code “read coils”. The address of the first coil to read is a 16 bit and the number of coils to read is 16 bit. The value of each coil is a binary, which is 0 for off, and 1 for on.

-006A is the Data address of the requested register. The range of the addresses of the analogue holding register is from 40001 to 49999. The 006A in decimal is106, which corresponds to 40107.

-0002 is the number of requested register, which range from 40107 to 40108. The total number is 2 registers.

-7687 is for the error checking for Modbus, the Cyclic Redundancy Check (CRC)

The mapper class after the packet has been divided as seen above, the keys are sent to the shuffle and sort and the state will be update. The shuffle and sort will clearly state the parameters that are expected to be in each part of the packet such as the Slave ID after the keys have been collected and grouped together. Identifying anomalies will be easy here since the expected parameters are clearly stated in the code. This will show the normal state and anything that falls outside it is not normal. In order to further detect anomalies in the system, the MapReduce job will compare the current state of the system with the last saved in the batch in order to detect anomalies such as train collision. If the last saved train barrier that was saved and the current train are on the same track, there is a possibility of collision and that should be reported as an anomaly. The reducer class on the other hand will assemble the data together, report any detected anomaly and update the Time To Live (TTL) of the group and if more than one group detected anomaly, update the global TTL as seen in figure 3.

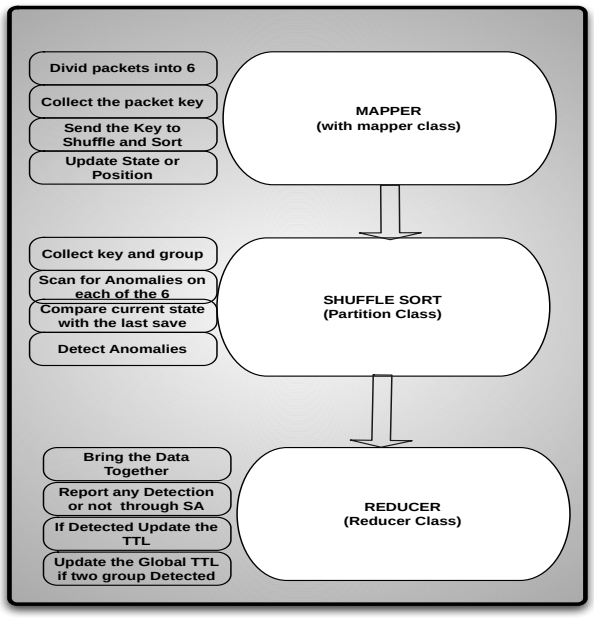


Figure 3: MapReduce/Serving Layer

Speed Layer

The speed layer (figure 4) works with the real time data such as on and off which can be open and close the barrier of a train system. This can be classified and near real time analysis of the activities of the system. The MapReduce in the previous section will analyse the saved data as well as the real time data and this code will be implemented in the speed layer. Spark was selected because it is suitable for real time analysis of data, Ballou, 2014. Apache Spark is fast and it is suitable for large scale data processing compared to Apache Storm. Spark can operate as a standalone master/slave processes which does not necessarily require Hadoop. Apache Spark has more popularity and stability compared to Storm, the development and contribution from the community cannot be compared to that of Storm. This shows that any issue with the Spark can be resolved faster compared to Storm and fault tolerance in Spark is better also.

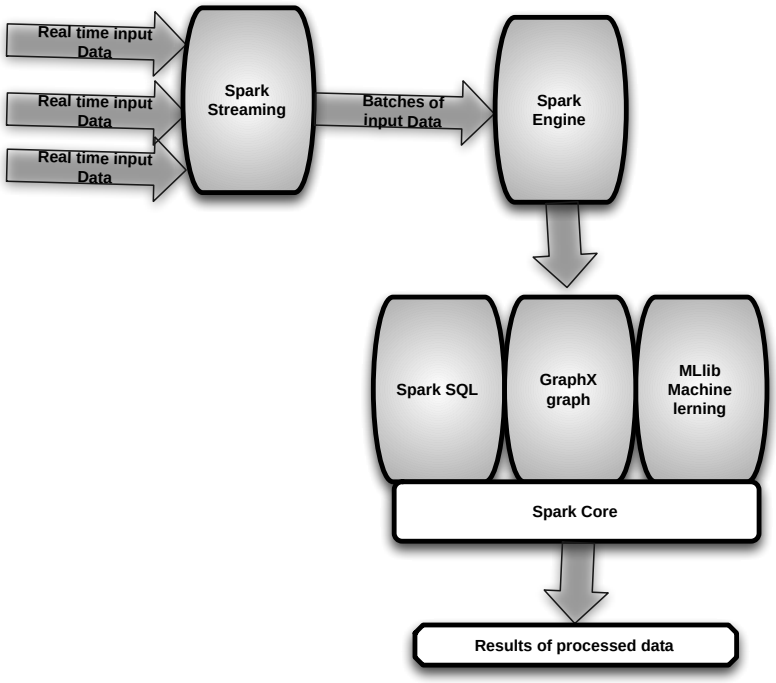
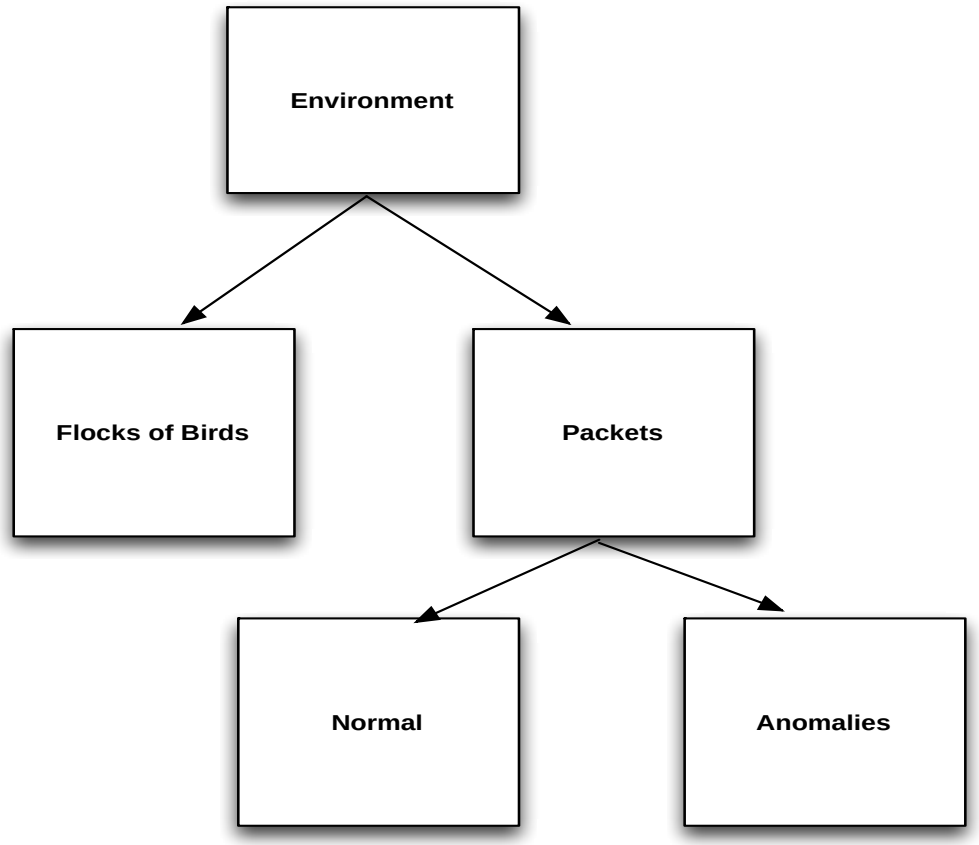


Figure 4: Speed Layer

Lambda architecture is suitable for applications that have some latency issue for data collection and availability (Kiran et al, 2015). Utilisation of lambda architecture and Apache Spark will help in minimising the latency issue that might be witnessed during detection processes. The data that are stored

in the batch layer will be valuable for any forensic investigation and it serves as a security for recovery of the system in terms of disaster. Spark as a streaming engine will serve for near real time detection of anomalies in the system.

The Algorithm Process



The System Design:

This is the whole system and what it needs in order to run. This involves the environment, flocks of birds' algorithm and the packets that would be analysed by the algorithm. Through the analysis of the packets, it will be deduced whether a packet is normal or anomaly.

Environment:

The environment here is the SCADA system, which is the physical equipment that can be added and removed from the system. This includes the PLC as well as the HMI that collect data from the each other. The flocks of bird's algorithm will reside in the environments. However, since the overall system architecture will be based on the No-Trust-Zone architecture according to Okeke and Blyth (2016), the

environment will be made up of separate hardware. The ALF will be made up of a separate hardware that will be integrated as part of a system operating in the cloud environment. This is the first module that will be programmed where the birds will be operating for analysing the packets.

Flocks of Birds

The second module that will be programmed is the flocks of birds'. There are characteristics that are very important in implementing flocks of birds algorithm and modelling it to fit the ALF. The followings are going to be integrated into the algorithm.

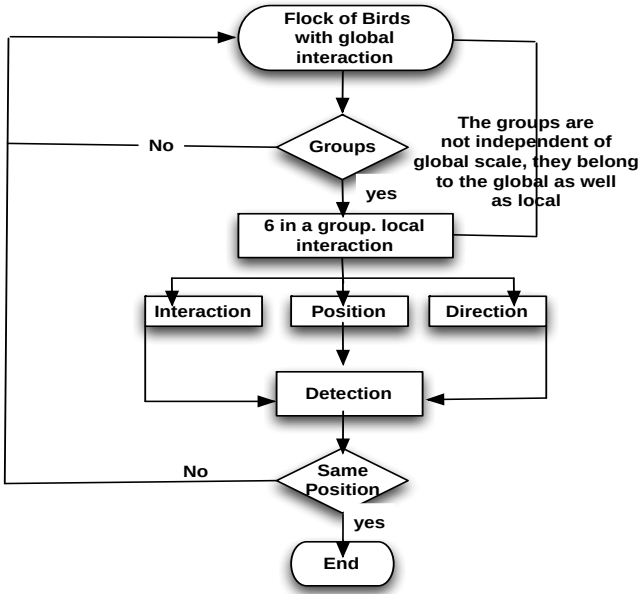
1. The algorithm will be initialised randomly and the initial number of birds will be between 1000.
2. The algorithm or the birds will be operating in an environment as stated above
3. The environment has cells and each cell can only accommodate maximum of six birds
4. The birds are following three flocking rules; separation, Alignment and Cohesion
5. The birds that are in the same cell are neighbours. This will be the way to get the number of neighbours and it will be used in the next approach
6. A cell that does not have up to six birds should not participate in data analysis
7. When something is detected, the TTL will be increased by 100% and when nothing is detected and the time to live finishes, the bird will die
8. If a bird in the group detects something, all the birds in that group will increase their TTL.
9. If two groups detect something within their TTL, new birds should be produces (reproduction) and that should make up 1000 in numbers as initially introduced.
10. The detection is based on the six birds approach and following MapReduce methodology according to Okeke and Blyth (2016)

Packets

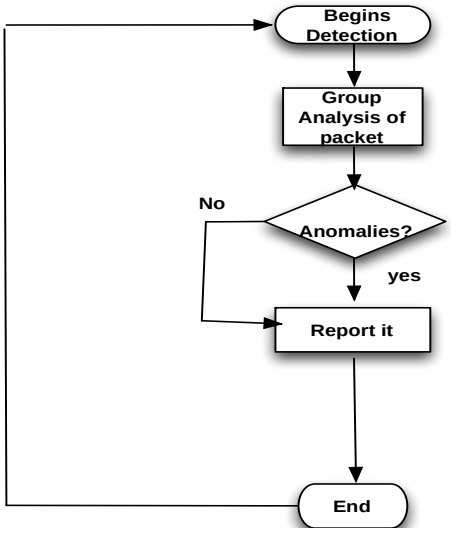
Packets here are the protocols such as ProfiNet and Modbus. These are the two protocols that will be used both in testing and experiments. The train system developed for the purpose of this experiment has only two different types of PLC; Siemens S7 1200 and Allen Bradley. The S7 from Siemens uses ProfiNet protocol while Allen Bradley uses Modbus for communication.

Designing The Algorithm:

The Flowcharts below are the structure of the algorithm and what it would address in terms of the detection by the birds as well as the structure of the predator (Anomaly Packets from the SCADA and Networks)



Algorithm Approach



Detection Approach

REFERENCES

- Marz N and Warren, J. (2013) Big Data: *Principles and Best Practices of Scalable Real time data systems*. Manning Publication.
- Derrick, H. (2009) The History of Hadoop: From 4 Nodes to The Future of Dta. Gigaom
- Amazon (2015) Lambda Architecture for Batch and Real-Time Processing on AWS with Spark Streaming and Spark SQL [online] Available at: https://d0.awsstatic.com/whitepapers/lambda-architecure-on-for-batch-aws.pdf. Last accessed 23.05.2016
- Kiran, M. Murphy, P. Monga, I. Dugan, J and baveja S. (2015) Lambda Architecture for Cost-effective Batch and Speed Big Data Processing: in Proceedings of 2015 IEEE International Conference on Big Data, Santa Clara, CA, PP.2785-2792
- Ballou, K. (2014) Apache Storm VS. Apache Spark [online] Available at: http://zdatainc.com/2014/09/apache-storm-apache-spark/. Last accessed 25.05.2015

Class Ingester Pseudocode

```
Import pcap, struct, happybase, dpkt, traceback, threading, socket, hashlib, datetime, ModbusParser, S7parser
class  Ingester(object):
define _ init_ (self, cloudIP, BirdsdetecHbaseIP, BirdTable): ##constructor
        HbaseIP = BirdsdetecHbaseIP;
        try connection to Hbase
                apply happybase connection using HbaseIP
                open the connection()
        print error message if the connection could not be established

        try connection to the table (BirdTable)
        print error message if connection could not be established

        if connection goes
        print "about to capture packets and information"
        initialise the pcap instance
        for timestamp, buf in pcap instance
        t = packetprocessor (buf, self.table, timestamp)
        run t ()  ## This will run the packet processor class
```

Code Snippet 1 Ingester

Packet Processor Class Pseudocode

(threading.Thread)

```
        define _ init_ (self, data, TableConn, timestamp): ##constructor
                initialise TableConn
                initialise timestamp
                initialise data
                set result to false
        define  insertHbaseRow method (self, RowKey, Data)
                try:
                        initialise HbaseRowKey and format date and time data
                        put the data and the HbaseRowKey in the tableConn
                else display exception
        define run method (self) ## this is running another thread
                use dpkt package to manipulate the Ethernet data
                initialise HbaseKeyHash = hashlib.md5
                update the source mac address
                update the destination mac address
                update the timestamp
```

##Test the type packet (for S7 packet)

Test if it is an ip packet

Test if it is a TCP packet

If the source or destination port is equal to 102

If the data length is greater than 110:

Use the dpkt package to manipulate the TCP data

Put the data in HbaseRow

Else if the TCP source or destination port is equal to 22

Treat it as a dns packet

Process the dns packet and insert into HbaseRow

Else treat it as a TCP data

##*For UDP data and packets process*

Else if the IP packet using the dpkt package is a UDP

Process it as UDP packet

If the source and destination port is equal to 22

Process it as DNS packet and insert into HbaseRow

Else if the source and destination port is equal to 67 and 68

Process it as DHCP packet and insert into HbaseRow

Else process it as UDP packet and insert into HbaseRow

##*Test the type packet (for Modbus packet)*

Test if it is an ip packet

Test if it is a TCP packet

If the source or destination port is equal to 502

Read register data at the address (0, 10)

Put the data in HbaseRow

##*For ICMP data and packets process*

Else if packet using dpkt package is a ICMP

Process it and insert into HbaseRow as ICMP

##*For ARP data and packets process*

Else if packet using dpkt package is a ARP

Process it and insert into HbaseRow as ARP

Define Ethernet:

Return source and destination mac address

Define ARP:

Return arp operation, source mac address, source ip address, destination mac and ip address

Define UDP:

Return source and destination udp port

Define TCP:

Return source and destination TCP port

Define DHCP:

Return dhcp operation

Define DNS:

Return dns operation

Define ICMP:

Return icmp type and code

Define S7Parser:

Return S7 packet

Define ModbusParser:

Return Modbus packet

Define merging dictionary (merge_dicts):

Return the result

Define mac address:

Return formatted mac address

Define ip address:

Return socket.inet_ntop(socket.AF_INET, address)

Define main():

Instance = Ingester (ip address here, detection class)

If_name_ equals 'main': main()

Code Snippet 2 Packet Processor

Pseudocode Cloud Interface

Getting a handle to the API Client first

From the cm_api_client which is cm_api package import ApiResource

Initialise the cm_host

Initialise the api (cm_host, username and password)

Getting all the list of the clusters and services

Reset CDH (depending on the version, eg 4 or 5) to original state

Get all the cluster:

Print all the cluster names

Get all the services:

Print all the services

Code Snippet 3 Cloud Interface

Pseudocode S7 Parser Class

Import dpkt

Import ipaddress

Import struct

Class S7Parser:

Construct S7 packet

Define initial packet

S7 packet = packet

Define the packet type by the type of PLC S7-1200 = 0x72 or S7-200-400 = 0x32

Return the number based on the type of PLC

Define the information to parse

Initialise control to 0

Return the value of Remote Operation Service Control (ROSCTR) second index after the packet type

If the ROSCTR value is equal to 1 or 7 ##it is a user data request for 7 and 1 is a request.

Return header byte from [0:10]

Increment control by 10

Unpack the header bytes ('!BBHHH', S7_header) ## ! = Network which is big-endian, B = unsigned char, H = unsigned short

Else if ROSCTR value is equal to 2 or 3 ## 2 = acknowledgement and 3 = response data

Return header byte from [0:12] ## 2 and 3 have 2 error bytes

Increment control by 12

Unpack the header bytes ('!BBHHH', S7_header)

S7 header is equal to unpacked S7 packet from [0:12]

If the header parameter in the fifth index [4] > 0 ## if there is a parameter

Then S7-parameter equal S7 packet controls plus S7 header in the fifth index

Control is equal to control plus S7 header in the fifth index

If the ROSCTR value is equal to 1 or 2 or 3

Function code is equal to integer in the first index of the S7-parameter encoded in hex

If the S7-header in the fifth index is greater than one## more than Function code, not finished yet, therefore do Awk

Count parameter byte in the second index in hex

Parameter size in the fifth index

If the header parameter in the sixth index [5] > 0 ## if there is a parameter

It's PDU bytes

If it has attributes and the count is greater than zero

Size is equal to parameter size or item count

If S7-header in the fifth index is greater than 2

If the value of the ROSCTR is equal to 7

Get the first 4 bits of the number

If parameter type equal 4

Get the size left by removing the 4 bits

Format the string to 13 byte single strings plus unsigned char

If the size left is greater than zero

Format the string and the size left

Get the parameter details by unpacking the format string and S7 parameter bytes

Else unpack the parameter details by unpacking the format string and S7 parameter bytes.

Parameter head in the detail in the first index

Sequence number in the detail of the index number 6

Count equal one.

Else if ROSCTR value is equal to 1 or 2 or 3

Current value is equal to 2

Set item to empty list

For I in range 0 to the number of the item

Unpack item based on the format '!BBBHHB3s'

New current equal to item size

Else if data length is greater than 1

Set item header to empty list

Set the content to empty list

Current is zero

If function code is not equal to 29, 30, and 31

The function does not have a data

Current is equal 4

If the item header is equal to 32

Get the list of the first item header

Get the other fist, which is 4

The length of the item is equal to first item and the other

If the 6th index is equal to 5 and the count is equal to 1

Item length equal 1

Format item length

Append item content when unpacked

Increment/update current

If current is equal to 1

Increment current by 1

If the 6th index of the header is equal to 0

If ROSCTR value is not equal to 2 and function code not equal to 29,30 and 31

Initialize the item address

Check from the range 0 to item count

Append the address the ith and 8th index encode in hex

Define packet details here:

Print the length of the packet

Print the hex of the S7 header bytes

Print magic number in hex ## for S7-1200 is 0x72 and for the older model is 0x32

If the ROSCTR value is equal to 1 or 3

If the function code is equal to 4 (Read var)

Print function code

Else if function code is equal to 5 (write var)

If 240

print: 0xf0 (set up communication)

If 29

Print: 0xid (start upload)

If 30

Print 0x1e (upload)

If 31

Print 0x1f (end upload)

If ROSCTR value is 1

Print (packet type: request job)

If 2

Print (packet type: Awk)

If 3

Print (packet type: response (Awk-data))

If 7

Print (packet type: user data)

If the S7 header in the fifth index if greater than 0

Print (S7 parameter area) format S7 parameter byte into hex

If ROSCTR value is equal to 1 or 2 or 3

If it has attribute (items)

For index, item in enumerate (self.items)

Print items and address. Format index and item in 8th index in hex

If ROSCTR value is equal to 7

Print (parameter head) format it and encode in hex

Print (sequence number) format sequence number

Else if the S7 header in the sixth index is greater than 0

Print (S7 PDU) format it and encode it in hex

If it has attribute (item contents)

For index, item in enumerate (self.item_contents)

Print (item, data, format it and encode in hex)

Else (print nothing)

Code Snippet 4 S7Parser Or ProfiNetPaser

Pseudocode ModbusPaser Class

Import the necessary libraries

Class ModbusPaser:

Construct the Modbus packet

Initialize the new instance of the decoder

Framer to use

Message encoder if the message needs it

Define decode

Try decoding the supplied message

Decode the message based on the frame (TCP)

Get the trasactionID

Get the protocolID

Get the DataLength

Get the unitID

Get the function code

Code Snippet 5 ModbusPaser

Pseudocode World Class /Environment

Import the necessary modules such as math

Class world:

Construct the world using the dimension height and width

Define get height

Return height

Define get width

Return width

Define get item for the bird at a given position

Return bird at the position

Define set item for a position to contain bird

Set the position if it within the world dimension

Define remove item from the position

Remove bird from the position (disable activity or mute the bird)

Define neighbors

Return the list of first six birds within distance (in a cell)

Define group

Six birds is equal to one group

Count is equal to groups

Group is equal to zero

Count is equal count plus one

Return groups

Define moveAllBirds a step forward

For bird in birds

Bird.step()

Code Snippet 6 World Class /Environment

Pseudocode Cell Class

Import all the necessary modules

##global constants

Global height based on the height of the World class

Global width based on the width of the world class

Class Cell:

Construct the cells using height and width (2D) of the world class

Grid = [for n in range (height)] for n in range (width)]

Initialize cells number from 1 to 6

For birds in range (number of birds):

X = randint (width, height)

Y = randint (width, height)

If grid[x][y] contains birds = = false

Move to the next

If grid[x][y] contains birds and the number = = 6

Set up a counter and count it

Qualifies for incoming packet analysis

Count cell numbers from 1 to 6

Pos1 = 1

Pos2 = 2

Pos3 = 3

Pos4 = 4

Pos5 = 5

Pos6 = 6

Define Time-To-Live for each bird

If bird Time-To-Live finishes

Remove bird (mute)

Define neighbors

Only birds in one cell

Maximum of six in a cell

Define group

A group equal neighbors

While group equal false

Continue checking the cells

Increment counter when found

Code Snippet 7 Cell

Pseudocode Boids Class

Import all the necessary modules (ModbusParser, S7Parser, Detection, Vector, Flocking, Cell)

constructor for the Boids class in a random position in the world

initialize the world and its dimension (x,y)

```
            get the width and the height
        initialize the position based on the vector (x,y)
            get the position
        initialize the velocity ##they fly within the world
        initialize their movement as up, down, left and right

Define move:
        move the birds to a new position in the world and cell
        get the width and height of the world
        determine the new position using their current position and velocity for x and y
        if x and y are new position and its free
            move to new position
            delete the old position
Define step:
        implement the three rules with the individual velocity and weight
            rule 1 = avoid collision
            rule 2 = match speed with the neighbors
            rule 3 = move to the center
        check individual velocity and the new velocity
        move ()
##the next is to get the neighbors based on the cell
Define neighbors:
        ##Return the list of birds in every cell
        Neighbors = self.cell.neighbors(position, cell)
        Declare empty list of neighbors in a cell
        For boid in neighbors:
            Get the neighbors in the same cell
        Return the list of birds
Define avoid collision ##rule 1
        Get the visible neighbors based on the cell inmates
        If no neighbors
            Return zero vector
        Else return the average position based on the cell

Define match speed ##rule 2
        Get the number of visible neighbors based on their cell
        If no neighbors
            Return zero vector
            Else return the number of neighbors I the cells
Define move to center
```

Get the number of visible birds based on the cells

If statement as in the avoid here

Move in the to another cell

Code Snippet 8 Boids

Pseudocode Pair Class

Import all the necessary modules

Class pair:

Constructor pair with the parameter a and b

Initialize a and b to be the first and second pairs

The parameter a will check with while loop through for a cell with complete birds (6)

The parameter b will be the next location of the cell with six birds

Define add with a counter

Return the addition of the first, second and other cells

Increment the counter

Define subtraction

Return the removal from the count as the number reduces

Decrement the counter

Code Snippet 9 Pair

Pseudocode Vector class

Vector class:

Constructor object to initialize the vector class for x and y

Define add

Return self-vector and others in the cell

Get item for self and index

Return the self-index value of the coordinate

Set item for self and index

Return the index coordinate value of self

String of self-value and index

Return the self-value of x and y

Magnitude of self

Return the length of self

diffAngle of self and vector 2

return the self-angle and vector 2

unit of self

return the unit vector in the same direction

Code Snippet 10 Vector

Pseudocode Flocking class

Import all the necessary modules

Set the width and height of their world

Set the number of birds

Define the main class ()

    Set turtle from the turtle Module () ## for testing purposes only

    Get the screen () ##for testing purposes only

    Set their world (coordinate of the width and height)

    Set the number of birds ()

    Set the turtle to hide ()

    Set sky to be equal to world dimension

    For index in range (number of bird)

        Bird = Boids(sky)

    While there are birds (bird)

        Sky.stepAll()

    Update the screen()

Main()

Code Snippet 11 Flocking

Pseudocode Detection Class

Import all the necessary libraries

Initialize the detection class and pass on the parameter (Modbus and S7)

Determine the packet type

If it is Modbus:

    Get Modbus packet structure (TransactionID, ProtocolID, DataLength, UnitID, FunctionCode)

        Return the structure

Else:

    Get the S7 packet structure (functionCode, PacketType, ROSCTR, DataLength, PacketHeader)

        Return the structure

Define check (numberOfBirds, cell, position, Modbus, S7)

    If the number is equal to 6 and the location of the cell from top left, the birds are free

        Proceed to detection

        Birds position in the cell

            1 = source IP

            2 = destination IP

            3 = data length

            4 = function code

            5 = verify function code

```
                6  =   verify with the backend for reliability
                If function code is not equal to (disallowed function codes)
                Continue to the next
                Check = check + 1
        If any anomalies if found in the packet
                Report it through SA
        The last bird verifies the last information saved on the HDFS
                If the gate opposite is open and there is a train
                        Flag anomalies to SA
Else if the number of birds in a cell is not equal to six
        Move to the next cell and continue search until the condition is met
Else continue checking for packets
```

Code Snippet 12 Detection

RESEARCH METHODOLOGY

Research methodology is an approach that will show how the problem identified in the literature could be solved (Kothari, 2004). The methodology is a driver that will help in explaining the methods as well as directing the research in arriving at the solution (Rajasekar, Philominathan, and Chinnatambi, 2013). However, if the right methodology is employed, it will surely yield the right findings and result (Liker, 2004 p.86)

This chapter will outline the procedures of the operation coupled with the ideas, methods and outcomes of the study at this stage. These are the set methods and procedures that were mapped out to achieve the objectives of the study. This involved the design of the study, which is like a map that will navigate the study into the desired solution.

The design encompasses some element such as approaches for the study. The preferred and selected method for this study will be critically evaluated. The reason for selecting this method will be given and justified. The identification of the research question and justification the hypothesis will follow based on the method that will be adopted. This will be followed by the tools or techniques that will be used in the approach. Their application in this study will be justified based on the results of the study. However, the data collection strategies that will be employed in this study will be explained. The reasons for using or employing such strategies will be justified based on the aim and objectives of the study. The process that will be followed in the collection of data will be explained and documented.

Research Method

During this study, number of research methods were considered but selecting the right method is the goal. Hence selecting the right method might involve asking the right questions. The core of this research is to model the collective behaviour seen in the flock of birds for the detection of anomalies on the Industrial Control Systems (ICS). There have been several models that would help in explaining this collective behaviour. Cavagna et al (2014) Modelled the behaviours using maximum entropy. Kattas, Xu and Small (2012), Ballerini et al. (2008), Couzin, (2009) through their various researches explained some of the collective behaviours seen in birds. When these animals are in group, they tend to be difficult to catch than a lone bird as hypothesised by Pulliam (1973) known as Many-Eyes hypotheses. Comparing evidences on these emergent behaviours seen in the flocks of bird, especially their predator detection approach, some of the questions below arises;

1. How can this behaviour be emulated for securing ICS?
2. How can the approach be replicated in ICS?
3. What are the important aspects of this that would be viable in solving the current ICS challenges of huge amount of data or in other word big data challenges?

These questions will be answered using the models and integrating it into the ICS. The models will show a solution to big data problem for ICS. The method to be used in this study will involve designing and using a case study for data collection. The entire process will follow the design seen in figure 3.1 (Yin, 2009). The reason for using a case study in demonstrating and evaluation of this research will be given. The initial step based on the information in figure 3.1 is planning for the research. The process of planning begins by understanding the project and devising examining the initial hypothesis followed by the objectives. Based on the problems identified during the initial literature review, the whole research is to answer the questions that arose with some evidences or to solve the problems. Whether the hypothesis or the research questions, they have only one thing in common, solving the problems. Yin (2009) identified three important things in planning a case study as seen below;

- Identify research questions or other rationale for doing a case study
- Decide to use the case study compared to other methods of analysis.
- Understands its strengths and limitations

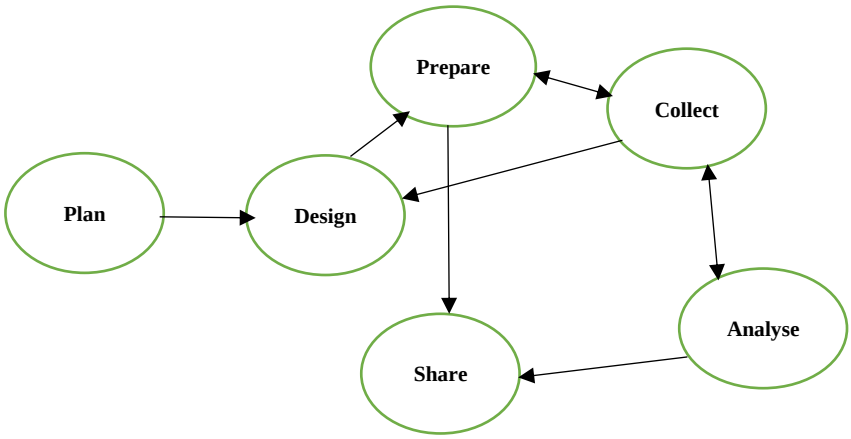


Figure 0-1 Research Planning Approach (source, Yin,2009)

Research Design

A research design is a map that shows the path to answering research questions or arriving at the objectives. It provides solution and makes it simple when collecting evidences that will assist in answering the research questions (Maxwell, 2000). There are principles and procedures that researchers have followed (Seale, 2006 P.130);

- Clear research question and a generated hypothesis
- The tools or method employed should yield robust data analysis so that the research aim will be met.
- Ethical issues should be considered when selecting and using research approaches.

Comparing the above three planning approach with the one from Yin in research meth section, both are communicating the same idea. They all pointed out the important of having a clear research question. Both highlighted the important of chosen the right method and know the reason for chosen a particular method. However, the later part highlighted ethical issue while Yin was on the understanding the strength and weaknesses of the method. This does not mean that his approach does not consider ethical issues, rather this was at the planning stage, while Seal was at the design stage. Hence, both approaches recognised the importance of considering ethical issues. The methodology for this research will be Design Science research (DS) and the approaches will be a combination of case study and exploratory approach. This will involve a single case study and exploring literature and experiment of the current and future solution based on the research question. According to Yin (2009) employing two methods such as case study with exploratory or experimental approach will yield a robust result. The hypothesis that has been followed for this study up till now is;”

Research Hypothesis

"*Predator prey model approach can enhance the detection of anomalies on SCADA industrial process control systems"*

The hypothesis arouses many questions in the mind, because of the subject of this research. Birds are animal that flocks in thousand and even millions based on the information in the literature review in chapter 3. These are animals and not a computer hardware or software that can easily be integrated into computer system or ICS. The next section will begin by considering some of the questions that were earlier stated and as well as new ones.

Research Question Identification

The research question will determine how the research will flow and the outcome of the research, Zucker (2009). Research question should be guided by the problems identified during the initial literature review and the very purpose of the research. The questions should be influenced by what the research is trying to learn from the study such as; why is this research being conducted? What is the purpose of the research and what would be learned from the research? What knowledge are you trying to pass on from the research? (Yin, 2009)

The primary aim of this study is to explore, experiment and explain through models the use of artificial life in detection of anomalies on ICS. This study will focus on modelling the collective behaviours seen in flocks of birds, such as detection of predator in such a large group. Determining the object of this study is very important because it will help in formulating the question according to Soy, (1997). This will help in focusing the research around the object in order not to deviate from the aim. The objects of this research are the flocks of bird's detection approach, the SCADA systems and the big data. When the focus of the study is determined, a methodological approach that will guide in the study of this complex phenomenon will follow (Yin, 2009; Soy, 1997). Based on the aim of the study, the main subject of this study is the detection approach by the flocks of birds or flocks of bird's approach in detecting predator. Emulating this approach in detecting anomalies in ICS would be a great contribution to knowledge. The problem is how can the approach be replicated on the ICS? This will form the questions for this study as seen below.

9 How would the detection approach seen in the flocks of bird be modelled for anomalies detection on ICS?

10 How can this approach solve the problem of huge amount of data being generated from sensors?

11 How can this model be compared to other detection approaches such as Snort rules and models such as Artificial Bee Colony (ABC) and Ant Colony optimisation (ACO)?

12 Where in the SCADA systems can the model be applied?

13 How would the anomalies be detected on SCADA systems data using this approach?

Chosen the Right Method

There are quite a few available research methods one could choose from, such as; Qualitative, Quantitative, Design Science (DS), Experimental as a method, Survey as a method, Case Study as a method and among others (Yin,2009 P.8). The adaptation and application of a research method is based on its suitability in solving the problem. Qualitative is applied when an in-depth investigation is needed using tools such as questionnaires, interviews and focus group. Quantitative is applied when measurement is needed in terms of numbers in quantity and it involves large amount of data or measurements (Shield and Twycross, 2003). Design Science research methodology is mostly on producing an artifact (Rossi and Sein, 2003). This requires some processes such as; identifying the problems and the motivations, successfully identifying the objectives that would yield the solution, designing and developing the product, demonstration, evaluation and communication of the study. Demonstration and evaluation can be performed through developing a case study to suit the study (Peffers, et al, 2007). Yin (2009, P. 8) explained that both experiment, survey and case study can be determined through the type of research questions. The questions developed in experiment are mostly how and why and requires the control of event behaviour. Survey requires no control, but the research questions are mainly; who, what, where, how many and how much? Case study has questions such as how and why but does not require behavioural control compared to experiment.

However, the questions in research question identification section are based on the initial knowledge of the problem and the system gathered in literature. Farrugia, et al (2010) pointed out that research question should be hypothesis driven. Meaning that around the hypothesis arises some questions that would be answered by drawing up some objectives. Hence objectives are the product of questions that arises from the hypothesis. In research design section above is the hypothesis and the more refined questions that arose from this are in research question identification section compared to question in research method section. The objectives will help in answering the research questions. However, before looking at refining

the objectives that would help in answering the research questions, it would be good a practice to critically analyse the generated research questions. Hulley, et al (2007) highlighted that a good research questions will produce a good research. Hulley and colleagues developed a criterial for a good research, the criterial that could produce a good research project are dependent on the following; Feasibility, Interesting, Novelty, Ethical and Relevant (FINER), figure 0.2.

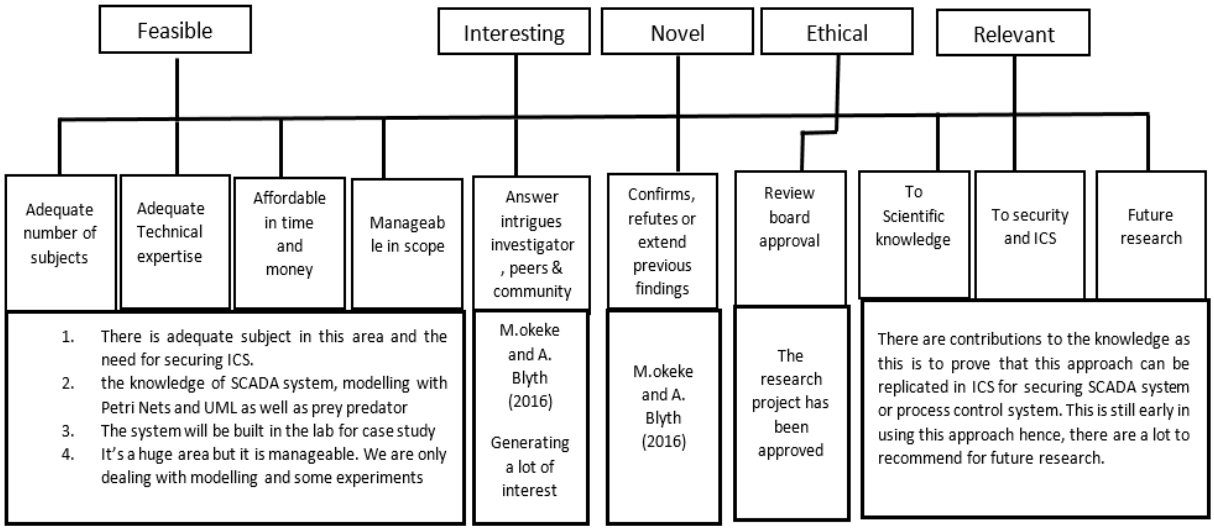


Figure 0-2 Criterial for a Good Research Questions

The FINER criteria showed that the research questions in research question identification section will produce a good research project. The objectives should be clear on how the research questions would be answered (Hulley, et al, 2007). This should be an active statement that is simple and easy to follow. Considering all the questions that were generated, below are the objectives that will be used in answering these questions or arriving at the hypothesis;

Research Objectives

1. To carry out research on the possibility of using the flocks of bird's approach for anomalies detection on ICS.
2. Carry out research on the possible best way to model the approach for the detection of anomalies on ICS
3. Carry out research on the best possible solution for big data
4. To create a SCADA test rig upon which a predator prey model can execute and to create a predator prey model. The test rig will facilitate the collection of initial data from the PLC and HMI that will be fed into the model in order to determine the anomalies.
5. To create the models and evaluate it for performance in order to validate the hypothesis.

6. To compare and contrast the result in order to prove or disprove the assertion that a bird of prey model can enhances the detection of anomalies on Industrial Control Systems.

The objectives above are simple, measurable, unambiguous and clear (Rojon, & Saunders, 2012). The objectives show how the research questions would be answered and or how to arrive at the research hypothesis. In order to know the next steps to follow, the journey so far will be constructed. Figure 3.3 is the diagrammatic representation of the journey so far.

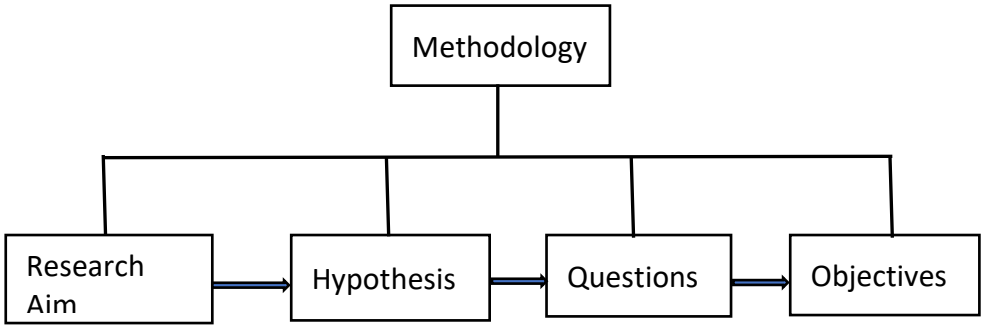


Figure 0-3 The process of developing the right method

The Research Method and The Reason for the Choice

A number of research methods were named in Chosen the Right Method section and their approaches were briefly explained. This study will follow the Design Science research methodology (DS) for modelling behaviours. Simon (1996, chapter 5) named it science of design which has its origin from engineering and the study of artificial. It is a discipline with the purpose and objective of solving problem through the creation of knowledge. When knowledge is created, problems can be solved through the creation of artifacts (Hevner et al, 2004). Hevner et al also pointed out that the DS research methods mainly focused on the creation and evaluation of artifacts. The artifacts can range from construct, design theories, Instantiation, methods and models. Below are the full definitions of what artifacts are based on, (Gregor and Jones, 2007; Hevner et al 2004; March and Smith, 1995). **Construct** is a mental concept based on symbols and vocabulary of an area of study or research. **Design theories** are the principles that govern the design of a system of study. **Instantiation** is about the implementation and prototyping of a system. **Methods** are the steeps in achieving the desired goal or in order word, planned goal. **Model** shows the representation of the proposition in relation to the construct. Gregor and Hevner added design principles, architecture and framework in the output of their DS research knowledge artifact. But as Gregor and Hevner

(2013) said, it should be complementary in order to increase knowledge than opposing each other. What is not in the other ones can be added in order to make it widely accepted.

Hevner et al (2004) elucidated that DS research main focus is about building and evaluation of artifacts for the purpose of meeting the discovered needs. They also explained that the effectiveness of a given artifact are primarily evaluated by the mathematical, computational and empirical methods. This will show a key contribution to the knowledge and methodologies. Based on the information from research method section till this point, research has shown that artifact are very important for DS research. It has also shown that model is an artifact as well as construct and methods. However, the selection of the DS research method among all the methods must be justify and the justification will begin by answering the questions on table 3.2 based on criteria on table 0.1, adapted from Ferris, (2009)

Dimension	Code	Possible Categories		Number
Desiderata	D1	Develop the theory of the field		3
	D2	Develop the practice of the field		
	D3	Develop the theory and the practice of the field		
Relationship to knowledge	K1	Knowledge is to be desired as the goal		3
	K2	Knowledge is desired for practical application		
	K3	Both knowledge and application to be desired		
Person who benefits	P1	Researcher		2
	P2	Others		
View of certainty of knowledge	C1	Knowledge is certain and absolute		2
	C2	Knowledge is relative and contingent		
View of tradition	T1	Conform to general pattern of discipline		2
	T2	Iconoclasts – challenge or reject tradition		
Objective of life	O1	To enjoy knowing	Life of leisure	3
	O2	To enjoy practice	Life of business	
	O3	To enjoy both knowing and practice	Life of leisure and business	

Table 0-1 Taxonomy of research methods (adapted from Ferris, 2009)

Dimension	Questions	Outcomes of the question
Meta-dimensional questions	What is the subject matter of the proposed project?	Distributed detection system in big data environment.
	Why will the proposed project be done?	To contribute to knowledge as well as obtaining a PhD
	Who will do the proposed project?	The PhD candidate
	For whom will the proposed be done?	Security community and the examiner
	When are the results of the proposed project be required?	The thesis is required by March 2018
	Where will the proposed project be done?	University of South Wales
Desiderata	D1 - Is the proposed project intended to make significant contribution to the theory of the field?	No, because no new theory is being developed.
	D2 - Is the proposed project intended to make significant contribution to the practice of the field?	Yes, because new idea and knowledge is being developed
Relation to knowledge	K1 - Is the knowledge expected in the proposed project primarily desired for its intrinsic value?	No, this is an empirical as well as applied research that will be further developed
	K2 - Is the knowledge expected in the proposed project primarily desired for its instrumental value as means to achieve something else?	Yes, as a means to achieve performance in the field of security
Person who benefits	P1 - Is the primary beneficiary expected in the proposed project the researcher?	No, the researcher and the security community, academia and organisations
	P2 - Is the primary beneficiary expected in the proposed project people other than the researcher?	Yes, the product is expected to be developed further by the developers.
View of certainty of knowledge	C1 - Does the proposed project presupposed that the knowledge to be developed concerns matters which objectively exist?	Yes, the project integrates into the system an improved effective knowledge.
	C2 - Does the proposed project presupposed that the knowledge to be developed concerns matters which are constructs of the community?	No
View of tradition	T1 - Does the proposed project presuppose that the existing framework of the field should be used as a foundation?	Yes, in the sense that anomaly detection is done using IDS. No, in the sense that this will challenge the existing knowledge

	T2 - Does the proposed project presuppose that the existing framework of the field should be rejected or vigorously challenged?	No

Table 0-2 Questions to determine the rightness of the method (adapted from, Ferris, 2009)

The Reason for DS Research Method

Examining the questions above in table 0.2, the design science method fulfilled the requirements based on the outcome of the questions that were asked. The questions were to determine the right method to be applied for achieving the right solution. Hence table 0.1 showed that design method is (D2|D3,K2|K3,P2,C1,T1,O3). However, O3 does not apply, although the knowledge will definitely be enjoyed by the researcher as well as the community. It will surely make life easier if well implemented in security for the detection of anomalies in the process control system. The DS research method is a methodological approach in solving problems by representing it through artifacts. Representing a problem in a way that the solution would be easily detectable (Simon, 1996, P.132). This study recognises the semantics of Design Research (DR) and Design Science Research (DSR). These two representations are used interchangeably in most research studies. The DR is mainly the study of designs, while the DSR is the newer formulation that includes design as a research method (Vaishanav and Kuechler, 2007). Vaishanav and Kuechler explained the important of well-developed research that will have a lasting impact. They stressed further that some research do not produce a lasting impact because their output maybe just be application without a good background, a working application. A research that will have impact can start with conceptual model contribution. Based on the research example given that went through a successful PhD, the artifact was a model representation of the solution which another student picked up from where it stopped. According to them, it was successful, and it is still having an impact because it has a sound background.

Based on the information gathered so far on the research hypothesis, questions, objectives and the selection of the methods. The questions and hypothesis were carefully selected based on the knowledge gathered during the review of literature. The study successfully identified the subject of this study and the methodology that will be used in modelling the approach in ICS. bringing all this together, what is needed is the guideline for a successful modelling of the concept. This study will adopt the DS research guidelines from Hevner, (2004). The table 0.3 are the seven guidelines that have been successfully applied in DS research.

Index	Guideline	Description of the Guideline	Answer
1	Design as an Artifact	DS research must produce a viable artifact in the form of a construct, model, a method or an instantiation	This research will produce models.
2	Problem Relevance	The objectives of DS research is to develop technology based solution to important and relevant business problems.	The artifact will solve the problem of distributed detection.
3	Design Evaluation	The utility, quality and efficacy of a design artifact must be rigorously demonstrated via well executed evaluation methods	The model will be evaluated based on functionality and performance,
4	Research Contribution	Effective DS research must provide clear and verifiable contribution in the area of the design artifact, design foundation and/or design methodologies.	The proposition and the models are the contributions
5	Research Rigor	DS research relies upon the application of rigorous methods in both the construction and evaluation of the artifact.	Using models in explaining a concept that has never been used before
6	Design as a search Process	The search for an effective artifact requires utilising available means to reach desired ends while satisfying laws in the problem environment	"Design is essentially a search process to discover an effective solution" (Havner et al 2004)
7	Communication of Research	DS research must be presented effectively both to technology oriented as well as management-oriented audiences	This will be presented to the examiners and in conferences

Table 0-3 Design Science Research Guidelines (Adapted from, Hevner et al, 2004)

The seven guidelines in table 0.3 shows that the study is in the right direction as all the seven guidelines were considered. The guideline will guide the research in terms of the requirements and steps that are needed to be implemented. Before delving into the modelling of the artifact, the environment will be defined and studied. Understanding the environment will make it easy in understanding the problems in the environment. This study will use a case study "Train system or Locomotive in Wales (Arriva)" as a SCADA system environment or as a testbed for this study. The only reason for this is that the Train system is the only SCADA system available at the moment for this project.

Experiment Definition

The experiments definitions will begin with the modelling of the whole system and idea in order to test the hypothesis. This will begin with designing the whole testbed as well as the bird model in order to understand the behaviour of the system. Using a UML diagram, the class diagram and sequence diagram, the system will be designed that will show the system, data flow and most importantly for the requirements gathering. Petri Nets formalism will be used for the modelling and testing of the system also for the case study, just to show the model in another light. Below are the steps that will be followed for the modelling and experimentation of the whole idea;

1. Data will be collected from the Train system in order to check the latency introduced as a result of sending it to the cloud for storage.
2. The data will help in identifying a pattern for the protocol's arrivals
3. The data will be used to model the communication protocol that will be used in detection
4. Requirements gathering for the modelling of the system will be carried out using UML diagrams

After the data has been collected and a pattern has been established as well as the model for the communication protocol, the next stage will be the modelling of the system. The initial model using UML diagrams will be for the virtualisation and information gathering of the system. The information will help in further modelling of the case study using coloured Petri Nets (CP)

1. The initial model of the Artificial Life Framework will be designed and modelled.
2. The Railway system will be modelled with trains on it using CP as a case study
3. The communication between the railway system and the server will be indicated. The model will show how data is being sent to Hadoop in the cloud and stored in HDFS and HBase.
4. To build the testbed and do the test of the program.
5. This is going to follow a walkthrough approach. The model will be modelled and tested
6. The model will operate on the data that normal IDS does not see such as Modbus and ProfiNet.
7. The model can pattern match on multiple data stream. Snort only see data such as TCP packet, but this model can match data from multiple data streams.

After the railway system has been modelled as well as its communication with other systems such as cloud system for data acquisition and storage. The next will be to model and construct the birds approach in detecting predator. This is the anomaly detection approach that will be adopted for this project which will be demonstrated in chapters 4 and 5.

1. It was assumed that the detection approach from the flocks of birds is based on the scanning frequency. The more the number of birds the less scanning and vise vasa. However, our approach is based on six birds in a group watching one another's movements, which is based on anisotropic as seen in literature review. Hence the model is based on these parallel movements, because detection is done concurrently which is
   Detection D = Bird 1 + Bird 2 + Bird 3 + Bird 4 + Bird 5 + Bird 6 = Substance S
   D = $\frac{S}{6\ Birds}$
2. The Birds will be modelled using the same UML and CP, for six birds approach
3. The detection approach will be modelled for anomaly detection on SCADA system
4. The location of detection model in the system will be modelled

Testing and evaluation

1. The data collected from the SCADA system will be compared with the one collected from the model and a recommendation for improvement will be made.
2. The models will be tested for performance. This is for the case study that was designed for this study.
3. The model can be compared against other IDS such as Snort rules and models such as ABC and ACO
4. The model operates on protocols such as ProfiNet and Modbus while other models does not see such protocols in the system.

Unit of Analysis

This aspect of this research deals with defining the case in a case study according to Yin (2009). It is about focusing on a particular aspect of the research for information retrieval in answering the research questions. The case study in this research is developed using a Train or Locomotive system. the Train system is developed with all the ICS equipment for monitoring and for data collection. However, the security of this system is paramount, therefore the research focuses on adopting the collective behaviours seen in the flocks of bird prey in detecting predator for securing the system. This study deals with two particular studies; the Train or locomotive

system which implements SCADA system as well as the flocks of bird prey approach in detecting predator. Hence, this case study will focus on the Train system and the unit of analysis is on data from the system.

The data that will be collected here are the data from the sensor, data from the Siemens PLCs and Schneider Electric PLC as well as data from the three HMIs. The data from the PLCs will be to determine the state of the sensors as well as the location of the locomotive. This will help in determining the sensor that is responsible for a possible collision of locomotives. This information will be feed into the model for the detection of anomalies in the system. however, for the HMIs, the data will be to determine the protocol; whether it is a Modbus data or ProfiNet. Collection of this data will help in building and constructing a pattern or structure of the data for the detection purposes. The structure will be what the birds will be looking out for and each bird will have something to analyse.

Summary

The information from the introduction of this chapter to this point has proven that DS research methodology will accomplish the task and arrived at satisfactory results. The hypothesis arouses some question that formed the research question. Prasad, Rao and Rehani (2001) explained that research question is a hypothesis in form of a question. They also emphasised that hypotheses are more firm compared to research questions that can be more in numbers. Hence, questions were discovered in order to narrow and simplify the hypothesis in this research. These questions are instruments that will be used in scanning the hypothesis in order to leave no stone untorn. Through the research questions, the objectives were formulated that will help in answering the questions.

The tables of questions in table 0.1 and 0.2 as well as table 0.3 was to verify the suitability of DS research method. The information proved that DS is the right method that will navigate the research to the desired destination. Figure 0.2 shows the viability of this research project and what it will mean to the community in the future. The experimental definition in section above is to identify how the system will be modelled as well as data collection. The modelling will start with identifying the communication among various variables in the system. How the birds in the system will communicate with the protocol as well as their environment. Hence, some approaches and methods for the modelling was devised by first using a UML for modelling the whole communication system and then proceed with Petri-Nets formalism for case study.

Testing, analysis and evaluation approaches for the model were identified. The data will be collected from train system in order to identify a pattern that will be used in modelling the approach. The model will be evaluated based on performance and against other models as stated in section 3.3.1 and the unit of analysis was identified in order to narrow the research focus and not to deviate from the aim of the research.

The problem with the current Intrusion Detection Systems as discovered in the literature are; Big data challenges; Snort and some other IDS do not operate on big data. Many of them do not take on multiple data stream. They are only applied to a single data stream. Our model can take on large volume of data because it is distributed and scale very well.

Train System Simulation

From the report below, the highest number of trains in the queue is 6 at steps 17 and 18 of the simulation. The numbers on the left side are the steps as explained in chapter 7 of this project. The information below is to show what was explained in chapter 7 with regards to different measurements. The simulation was carried out under different measurement and adjustments of the model and time. However different results were obtained under different approaches. Some approaches yielded lesser queue than others as explained. Below are three different simulations under different settings.

Simulations 1

```
CPN Tools simulation report for:

1       0       TP1 @ (1:Queue_System)
 - t = Train2
2       0       Queue @ (1:Queue_System)
 - n = []
 - t = Train2
3       0       TP1 @ (1:Queue_System)
 - t = Train2
4       0       TP1 @ (1:Queue_System)
 - t = Train2
5       0       Queue @ (1:Queue_System)
 - n = [Train2]
 - t = Train2
6       0       TP1 @ (1:Queue_System)
 - t = Train2
7       0       Queue @ (1:Queue_System)
 - n = [Train2,Train2]
 - t = Train2
8       0       TP1 @ (1:Queue_System)
 - t = Train2
9       0       Queue @ (1:Queue_System)
```

```
 - n = [Train2,Train2,Train2]
 - t = Train2
10      0       TP1 @ (1:Queue_System)
 - t = Train2
11      0       TP1 @ (1:Queue_System)
 - t = Train2
12      0       Queue @ (1:Queue_System)
 - n = [Train2,Train2,Train2,Train2]
 - t = Train2
13      0       TP1 @ (1:Queue_System)
 - t = Train2
14      0       TP1 @ (1:Queue_System)
 - t = Train2
15      0       TP1 @ (1:Queue_System)
 - t = Train2
16      0       Queue @ (1:Queue_System)
 - n = [Train2,Train2,Train2,Train2,Train2]
 - t = Train2
17      0       Queue @ (1:Queue_System)
 - n = [Train2,Train2,Train2,Train2,Train2,Train2]
 - t = Train2
18      0       Comm @ (1:Queue_System)
 - n = [Train2,Train2,Train2,Train2,Train2,Train2]
 - t = Train2
19      0       Comm @ (1:Queue_System)
 - n = [Train2,Train2,Train2,Train2,Train2]
 - t = Train2
20      0       Comm @ (1:Queue_System)
 - n = [Train2,Train2,Train2,Train2]
 - t = Train2
21      0       Queue @ (1:Queue_System)
 - n = [Train2,Train2,Train2,Train2]
 - t = Train2
22      0       Comm @ (1:Queue_System)
 - n = [Train2,Train2,Train2,Train2]
 - t = Train2
23      0       Queue @ (1:Queue_System)
 - n = [Train2,Train2,Train2,Train2]
 - t = Train2
24      0       Queue @ (1:Queue_System)
 - n = [Train2,Train2,Train2,Train2,Train2]
 - t = Train2
25      0       Comm @ (1:Queue_System)
 - n = [Train2,Train2,Train2,Train2,Train2]
 - t = Train2
26      0       Comm @ (1:Queue_System)
 - n = [Train2,Train2,Train2,Train2]
 - t = Train2
27      0       Comm @ (1:Queue_System)
 - n = [Train2,Train2,Train2]
 - t = Train2
28      0       Comm @ (1:Queue_System)
 - n = [Train2,Train2]
 - t = Train2
29      0       Comm @ (1:Queue_System)
 - n = [Train2]
 - t = Train2
30      0       Comm @ (1:Queue_System)
 - n = []
 - t = Train2
31      5       TP1 @ (1:Queue_System)
```

```
 - t = Train1
32      5       TP1 @ (1:Queue_System)
 - t = Train1
33      5       TP1 @ (1:Queue_System)
 - t = Train1
34      5       TP1 @ (1:Queue_System)
 - t = Train1
35      5       Queue @ (1:Queue_System)
 - n = []
 - t = Train1
36      5       Queue @ (1:Queue_System)
 - n = [Train1]
 - t = Train1
37      5       TP1 @ (1:Queue_System)
 - t = Train1
38      5       Queue @ (1:Queue_System)
 - n = [Train1,Train1]
 - t = Train1
39      5       Queue @ (1:Queue_System)
 - n = [Train1,Train1,Train1]
 - t = Train1
40      5       TP1 @ (1:Queue_System)
 - t = Train1
41      5       Queue @ (1:Queue_System)
 - n = [Train1,Train1,Train1,Train1]
 - t = Train1
42      5       Queue @ (1:Queue_System)
 - n = [Train1,Train1,Train1,Train1,Train1]
 - t = Train1
43      5       Comm @ (1:Queue_System)
 - n = [Train1,Train1,Train1,Train1,Train1]
 - t = Train1
44      5       TP1 @ (1:Queue_System)
 - t = Train1
45      5       Comm @ (1:Queue_System)
 - n = [Train1,Train1,Train1,Train1]
 - t = Train1
46      5       TP1 @ (1:Queue_System)
 - t = Train1
47      5       Queue @ (1:Queue_System)
 - n = [Train1,Train1,Train1,Train1]
 - t = Train1
48      5       TP1 @ (1:Queue_System)
 - t = Train1
49      5       Comm @ (1:Queue_System)
 - n = [Train1,Train1,Train1,Train1]
 - t = Train1
50      5       Queue @ (1:Queue_System)
 - n = [Train1,Train1,Train1,Train1]
 - t = Train1
51      5       Comm @ (1:Queue_System)
 - n = [Train1,Train1,Train1,Train1]
 - t = Train1
52      5       Queue @ (1:Queue_System)
 - n = [Train1,Train1,Train1,Train1]
 - t = Train1
53      5       Comm @ (1:Queue_System)
 - n = [Train1,Train1,Train1,Train1]
 - t = Train1
54      5       TP1 @ (1:Queue_System)
 - t = Train1
```

```
55      5       Comm @ (1:Queue_System)
 - n = [Train1,Train1,Train1]
 - t = Train1
56      5       Queue @ (1:Queue_System)
 - n = [Train1,Train1,Train1]
 - t = Train1
57      5       Comm @ (1:Queue_System)
 - n = [Train1,Train1,Train1]
 - t = Train1
58      5       Comm @ (1:Queue_System)
 - n = [Train1,Train1]
 - t = Train1
59      5       Comm @ (1:Queue_System)
 - n = [Train1]
 - t = Train1
60      5       Comm @ (1:Queue_System)
 - n = []
 - t = Train1
61      10      TP1 @ (1:Queue_System)
 - t = Train3
62      10      TP1 @ (1:Queue_System)
 - t = Train3
63      10      Queue @ (1:Queue_System)
 - n = []
 - t = Train3
64      10      Comm @ (1:Queue_System)
 - n = []
 - t = Train3
65      10      Queue @ (1:Queue_System)
 - n = []
 - t = Train3
66      10      Comm @ (1:Queue_System)
 - n = []
 - t = Train3
67      10      TP1 @ (1:Queue_System)
 - t = Train3
68      10      TP1 @ (1:Queue_System)
 - t = Train3
69      10      Queue @ (1:Queue_System)
 - n = []
 - t = Train3
70      10      TP1 @ (1:Queue_System)
 - t = Train3
71      10      Comm @ (1:Queue_System)
 - n = []
 - t = Train3
72      10      Queue @ (1:Queue_System)
 - n = []
 - t = Train3
73      10      Queue @ (1:Queue_System)
 - n = [Train3]
 - t = Train3
74      10      TP1 @ (1:Queue_System)
 - t = Train3
75      10      Comm @ (1:Queue_System)
 - n = [Train3]
 - t = Train3
76      10      Comm @ (1:Queue_System)
 - n = []
 - t = Train3
77      10      Queue @ (1:Queue_System)
```

```
 - n = []
 - t = Train3
78      10      TP1 @ (1:Queue_System)
 - t = Train3
79      10      Comm @ (1:Queue_System)
 - n = []
 - t = Train3
80      10      Queue @ (1:Queue_System)
 - n = []
 - t = Train3
81      10      Comm @ (1:Queue_System)
 - n = []
 - t = Train3
82      10      TP1 @ (1:Queue_System)
 - t = Train3
83      10      Queue @ (1:Queue_System)
 - n = []
 - t = Train3
84      10      TP1 @ (1:Queue_System)
 - t = Train3
85      10      Queue @ (1:Queue_System)
 - n = [Train3]
 - t = Train3
86      10      Comm @ (1:Queue_System)
 - n = [Train3]
 - t = Train3
87      10      Comm @ (1:Queue_System)
 - n = []
 - t = Train3
88      10      TP1 @ (1:Queue_System)
 - t = Train3
89      10      Queue @ (1:Queue_System)
 - n = []
 - t = Train3
90      10      Comm @ (1:Queue_System)
 - n = []
 - t = Train3
```

Simulations 2

```
CPN Tools simulation report for:
/cygdrive/C/Users/okeke/Desktop/CPN Models/Queuing System another..cpn
Report generated: Wed Jan 31 12:33:05 2018

1       0       TP1 @ (1:Queue_System)
 - t = Train2
2       0       Queue @ (1:Queue_System)
 - n = []
 - t = Train2
3       0       TP1 @ (1:Queue_System)
 - t = Train2
4       0       Queue @ (1:Queue_System)
 - n = [Train2]
 - t = Train2
5       0       Comm @ (1:Queue_System)
 - n = [Train2]
 - t = Train2
6       0       TP1 @ (1:Queue_System)
 - t = Train2
7       0       TP1 @ (1:Queue_System)
```

```
 - t = Train2
8       0       Queue @ (1:Queue_System)
 - n = [Train2]
 - t = Train2
9       0       TP1 @ (1:Queue_System)
 - t = Train2
10      0       Queue @ (1:Queue_System)
 - n = [Train2,Train2]
 - t = Train2
11      0       TP1 @ (1:Queue_System)
 - t = Train2
12      0       Queue @ (1:Queue_System)
 - n = [Train2,Train2,Train2]
 - t = Train2
13      0       Queue @ (1:Queue_System)
 - n = [Train2,Train2,Train2,Train2]
 - t = Train2
14      0       TP1 @ (1:Queue_System)
 - t = Train2
15      0       Queue @ (1:Queue_System)
 - n = [Train2,Train2,Train2,Train2,Train2]
 - t = Train2
16      0       TP1 @ (1:Queue_System)
 - t = Train2
17      0       TP1 @ (1:Queue_System)
 - t = Train2
18      0       Queue @ (1:Queue_System)
 - n = [Train2,Train2,Train2,Train2,Train2,Train2]
 - t = Train2
19      0       TP1 @ (1:Queue_System)
 - t = Train2
20      0       Queue @ (1:Queue_System)
 - n = [Train2,Train2,Train2,Train2,Train2,Train2,Train2]
 - t = Train2
21      0       Queue @ (1:Queue_System)
 - n = [Train2,Train2,Train2,Train2,Train2,Train2,Train2,Train2]
 - t = Train2
22      1       Comm @ (1:Queue_System)
 - n = [Train2,Train2,Train2,Train2,Train2,Train2,Train2,Train2]
 - t = Train2
23      2       Comm @ (1:Queue_System)
 - n = [Train2,Train2,Train2,Train2,Train2,Train2,Train2]
 - t = Train2
24      3       Comm @ (1:Queue_System)
 - n = [Train2,Train2,Train2,Train2,Train2,Train2]
 - t = Train2
25      4       Comm @ (1:Queue_System)
 - n = [Train2,Train2,Train2,Train2,Train2]
 - t = Train2
26      5       Comm @ (1:Queue_System)
 - n = [Train2,Train2,Train2,Train2]
 - t = Train2
27      5       TP1 @ (1:Queue_System)
 - t = Train1
28      5       Queue @ (1:Queue_System)
 - n = [Train2,Train2,Train2,Train2]
 - t = Train1
29      5       TP1 @ (1:Queue_System)
 - t = Train1
30      5       TP1 @ (1:Queue_System)
 - t = Train1
```

```
31      5       Queue @ (1:Queue_System)
 - n = [Train2,Train2,Train2,Train2,Train1]
 - t = Train1
32      5       Queue @ (1:Queue_System)
 - n = [Train2,Train2,Train2,Train2,Train1,Train1]
 - t = Train1
33      5       TP1 @ (1:Queue_System)
 - t = Train1
34      5       Queue @ (1:Queue_System)
 - n = [Train2,Train2,Train2,Train2,Train1,Train1,Train1]
 - t = Train1
35      5       TP1 @ (1:Queue_System)
 - t = Train1
36      5       TP1 @ (1:Queue_System)
 - t = Train1
37      5       Queue @ (1:Queue_System)
 - n = [Train2,Train2,Train2,Train2,Train1,Train1,Train1,Train1]
 - t = Train1
38      5       Queue @ (1:Queue_System)
 - n = [Train2,Train2,Train2,Train2,Train1,Train1,Train1,Train1,Train1]
 - t = Train1
39      5       TP1 @ (1:Queue_System)
 - t = Train1
40      5       TP1 @ (1:Queue_System)
 - t = Train1
41      5       TP1 @ (1:Queue_System)
 - t = Train1
42      5       Queue @ (1:Queue_System)
 - n =
[Train2,Train2,Train2,Train2,Train1,Train1,Train1,Train1,Train1,Train1]
 - t = Train1
43      5       TP1 @ (1:Queue_System)
 - t = Train1
44      5       Queue @ (1:Queue_System)
 - n =
[Train2,Train2,Train2,Train2,Train1,Train1,Train1,Train1,Train1,Train1,Trai
n1]
 - t = Train1
45      5       Queue @ (1:Queue_System)
 - n =
[Train2,Train2,Train2,Train2,Train1,Train1,Train1,Train1,Train1,Train1,Trai
n1,Train1]
 - t = Train1
46      5       Queue @ (1:Queue_System)
 - n =
[Train2,Train2,Train2,Train2,Train1,Train1,Train1,Train1,Train1,Train1,Trai
n1,Train1,Train1]
 - t = Train1
47      6       Comm @ (1:Queue_System)
 - n =
[Train2,Train2,Train2,Train1,Train1,Train1,Train1,Train1,Train1,Train1,Trai
n1,Train1,Train1]
 - t = Train2
48      7       Comm @ (1:Queue_System)
 - n =
[Train2,Train2,Train1,Train1,Train1,Train1,Train1,Train1,Train1,Train1,Trai
n1,Train1]
 - t = Train2
49      8       Comm @ (1:Queue_System)
```

```
 - n =
[Train2,Train1,Train1,Train1,Train1,Train1,Train1,Train1,Train1,Train1,Trai
n1]
 - t = Train2
50      9       Comm @ (1:Queue_System)
 - n =
[Train1,Train1,Train1,Train1,Train1,Train1,Train1,Train1,Train1,Train1]
 - t = Train2
51      10      Comm @ (1:Queue_System)
 - n = [Train1,Train1,Train1,Train1,Train1,Train1,Train1,Train1,Train1]
 - t = Train1
52      10      TP1 @ (1:Queue_System)
 - t = Train3
53      10      Queue @ (1:Queue_System)
 - n = [Train1,Train1,Train1,Train1,Train1,Train1,Train1,Train1,Train1]
 - t = Train3
54      10      TP1 @ (1:Queue_System)
 - t = Train3
55      10      TP1 @ (1:Queue_System)
 - t = Train3
56      10      Queue @ (1:Queue_System)
 - n =
[Train1,Train1,Train1,Train1,Train1,Train1,Train1,Train1,Train1,Train3]
 - t = Train3
57      10      TP1 @ (1:Queue_System)
 - t = Train3
58      10      Queue @ (1:Queue_System)
 - n =
[Train1,Train1,Train1,Train1,Train1,Train1,Train1,Train1,Train1,Train3,Trai
n3]
 - t = Train3
59      10      TP1 @ (1:Queue_System)
 - t = Train3
60      10      TP1 @ (1:Queue_System)
 - t = Train3
61      10      Queue @ (1:Queue_System)
 - n =
[Train1,Train1,Train1,Train1,Train1,Train1,Train1,Train1,Train1,Train3,Trai
n3,Train3]
 - t = Train3
62      10      Queue @ (1:Queue_System)
 - n =
[Train1,Train1,Train1,Train1,Train1,Train1,Train1,Train1,Train1,Train3,Trai
n3,Train3,Train3]
 - t = Train3
63      10      Queue @ (1:Queue_System)
 - n =
[Train1,Train1,Train1,Train1,Train1,Train1,Train1,Train1,Train1,Train3,Trai
n3,Train3,Train3,Train3]
 - t = Train3
64      10      TP1 @ (1:Queue_System)
 - t = Train3
65      10      TP1 @ (1:Queue_System)
 - t = Train3
66      10      Queue @ (1:Queue_System)
 - n =
[Train1,Train1,Train1,Train1,Train1,Train1,Train1,Train1,Train1,Train3,Trai
n3,Train3,Train3,Train3,Train3]
 - t = Train3
67      10      TP1 @ (1:Queue_System)
 - t = Train3
```

```
68      10      Queue @ (1:Queue_System)
 - n =
[Train1,Train1,Train1,Train1,Train1,Train1,Train1,Train1,Train1,Train3,Trai
n3,Train3,Train3,Train3,Train3,Train3]
 - t = Train3
69      10      TP1 @ (1:Queue_System)
 - t = Train3
70      10      Queue @ (1:Queue_System)
 - n =
[Train1,Train1,Train1,Train1,Train1,Train1,Train1,Train1,Train1,Train3,Trai
n3,Train3,Train3,Train3,Train3,Train3]
 - t = Train3
71      10      Queue @ (1:Queue_System)
 - n =
[Train1,Train1,Train1,Train1,Train1,Train1,Train1,Train1,Train1,Train3,Trai
n3,Train3,Train3,Train3,Train3,Train3,Train3]
 - t = Train3
72      11      Comm @ (1:Queue_System)
 - n =
[Train1,Train1,Train1,Train1,Train1,Train1,Train1,Train1,Train3,Train3,Trai
n3,Train3,Train3,Train3,Train3,Train3,Train3]
 - t = Train1
73      12      Comm @ (1:Queue_System)
 - n =
[Train1,Train1,Train1,Train1,Train1,Train1,Train1,Train3,Train3,Train3,Trai
n3,Train3,Train3,Train3,Train3,Train3,Train3]
 - t = Train1
74      13      Comm @ (1:Queue_System)
 - n =
[Train1,Train1,Train1,Train1,Train1,Train1,Train3,Train3,Train3,Train3,Trai
n3,Train3,Train3,Train3,Train3,Train3]
 - t = Train1
75      14      Comm @ (1:Queue_System)
 - n =
[Train1,Train1,Train1,Train1,Train1,Train3,Train3,Train3,Train3,Train3,Trai
n3,Train3,Train3,Train3,Train3]
 - t = Train1
76      15      Comm @ (1:Queue_System)
 - n =
[Train1,Train1,Train1,Train1,Train3,Train3,Train3,Train3,Train3,Train3,Trai
n3,Train3,Train3,Train3]
 - t = Train1
77      16      Comm @ (1:Queue_System)
 - n =
[Train1,Train1,Train1,Train3,Train3,Train3,Train3,Train3,Train3,Train3,Trai
n3,Train3,Train3]
 - t = Train1
78      17      Comm @ (1:Queue_System)
 - n =
[Train1,Train1,Train3,Train3,Train3,Train3,Train3,Train3,Train3,Train3,Trai
n3,Train3]
 - t = Train1
79      18      Comm @ (1:Queue_System)
 - n =
[Train1,Train3,Train3,Train3,Train3,Train3,Train3,Train3,Train3,Train3,Trai
n3]
 - t = Train1
80      19      Comm @ (1:Queue_System)
 - n =
[Train3,Train3,Train3,Train3,Train3,Train3,Train3,Train3,Train3,Train3]
 - t = Train1
```

```
81      20      Comm @ (1:Queue_System)
 - n = [Train3,Train3,Train3,Train3,Train3,Train3,Train3,Train3,Train3]
 - t = Train3
82      21      Comm @ (1:Queue_System)
 - n = [Train3,Train3,Train3,Train3,Train3,Train3,Train3,Train3]
 - t = Train3
83      22      Comm @ (1:Queue_System)
 - n = [Train3,Train3,Train3,Train3,Train3,Train3,Train3]
 - t = Train3
84      23      Comm @ (1:Queue_System)
 - n = [Train3,Train3,Train3,Train3,Train3,Train3]
 - t = Train3
85      24      Comm @ (1:Queue_System)
 - n = [Train3,Train3,Train3,Train3,Train3]
 - t = Train3
86      25      Comm @ (1:Queue_System)
 - n = [Train3,Train3,Train3,Train3]
 - t = Train3
87      26      Comm @ (1:Queue_System)
 - n = [Train3,Train3,Train3]
 - t = Train3
88      27      Comm @ (1:Queue_System)
 - n = [Train3,Train3]
 - t = Train3
89      28      Comm @ (1:Queue_System)
 - n = [Train3]
 - t = Train3
90      29      Comm @ (1:Queue_System)
 - n = []
 - t = Train3
```

Simulations 3

```
CPN Tools simulation report for:
/cygdrive/C/Users/okeke/Desktop/CPN Models/Queuing System another..cpn
Report generated: Wed Jan 31 12:26:19 2018

1       0       TP1 @ (1:Queue_System)
 - t = Train2
2       0       TP1 @ (1:Queue_System)
 - t = Train2
3       0       Queue @ (1:Queue_System)
 - n = []
 - t = Train2
4       0       Queue @ (1:Queue_System)
 - n = [Train2]
 - t = Train2
5       0       TP1 @ (1:Queue_System)
 - t = Train2
6       0       TP1 @ (1:Queue_System)
 - t = Train2
7       0       Comm @ (1:Queue_System)
 - n = [Train2]
 - t = Train2
8       0       TP1 @ (1:Queue_System)
 - t = Train2
9       0       Queue @ (1:Queue_System)
 - n = [Train2]
 - t = Train2
10      0       TP1 @ (1:Queue_System)
```

```
 - t = Train2
11      0       TP1 @ (1:Queue_System)
 - t = Train2
12      0       Queue @ (1:Queue_System)
 - n = [Train2,Train2]
 - t = Train2
13      0       TP1 @ (1:Queue_System)
 - t = Train2
14      0       Queue @ (1:Queue_System)
 - n = [Train2,Train2,Train2]
 - t = Train2
15      0       Queue @ (1:Queue_System)
 - n = [Train2,Train2,Train2,Train2]
 - t = Train2
16      0       Queue @ (1:Queue_System)
 - n = [Train2,Train2,Train2,Train2,Train2]
 - t = Train2
17      0       TP1 @ (1:Queue_System)
 - t = Train2
18      0       TP1 @ (1:Queue_System)
 - t = Train2
19      0       Queue @ (1:Queue_System)
 - n = [Train2,Train2,Train2,Train2,Train2,Train2]
 - t = Train2
20      0       Queue @ (1:Queue_System)
 - n = [Train2,Train2,Train2,Train2,Train2,Train2,Train2]
 - t = Train2
21      0       Queue @ (1:Queue_System)
 - n = [Train2,Train2,Train2,Train2,Train2,Train2,Train2,Train2]
 - t = Train2
22      5       TP1 @ (1:Queue_System)
 - t = Train1
23      5       Comm @ (1:Queue_System)
 - n = [Train2,Train2,Train2,Train2,Train2,Train2,Train2,Train2]
 - t = Train2
24      5       TP1 @ (1:Queue_System)
 - t = Train1
25      5       TP1 @ (1:Queue_System)
 - t = Train1
26      5       TP1 @ (1:Queue_System)
 - t = Train1
27      5       Queue @ (1:Queue_System)
 - n = [Train2,Train2,Train2,Train2,Train2,Train2,Train2,Train2]
 - t = Train1
28      5       TP1 @ (1:Queue_System)
 - t = Train1
29      5       TP1 @ (1:Queue_System)
 - t = Train1
30      5       Queue @ (1:Queue_System)
 - n = [Train2,Train2,Train2,Train2,Train2,Train2,Train2,Train2,Train1]
 - t = Train1
31      5       TP1 @ (1:Queue_System)
 - t = Train1
32      5       TP1 @ (1:Queue_System)
 - t = Train1
33      5       Queue @ (1:Queue_System)
 - n =
[Train2,Train2,Train2,Train2,Train2,Train2,Train2,Train2,Train1,Train1]
 - t = Train1
34      5       TP1 @ (1:Queue_System)
 - t = Train1
```

```
35      5       TP1 @ (1:Queue_System)
 - t = Train1
36      5       Queue @ (1:Queue_System)
 - n =
[Train2,Train2,Train2,Train2,Train2,Train2,Train2,Train2,Train1,Train1,Trai
n1]
 - t = Train1
37      5       Queue @ (1:Queue_System)
 - n =
[Train2,Train2,Train2,Train2,Train2,Train2,Train2,Train2,Train1,Train1,Trai
n1,Train1]
 - t = Train1
38      5       Queue @ (1:Queue_System)
 - n =
[Train2,Train2,Train2,Train2,Train2,Train2,Train2,Train2,Train1,Train1,Trai
n1,Train1,Train1]
 - t = Train1
39      5       Queue @ (1:Queue_System)
 - n =
[Train2,Train2,Train2,Train2,Train2,Train2,Train2,Train2,Train1,Train1,Trai
n1,Train1,Train1,Train1]
 - t = Train1
40      5       Queue @ (1:Queue_System)
 - n =
[Train2,Train2,Train2,Train2,Train2,Train2,Train2,Train2,Train1,Train1,Trai
n1,Train1,Train1,Train1,Train1]
 - t = Train1
41      5       Queue @ (1:Queue_System)
 - n =
[Train2,Train2,Train2,Train2,Train2,Train2,Train2,Train2,Train1,Train1,Trai
n1,Train1,Train1,Train1,Train1,Train1]
 - t = Train1
42      5       Queue @ (1:Queue_System)
 - n =
[Train2,Train2,Train2,Train2,Train2,Train2,Train2,Train2,Train1,Train1,Trai
n1,Train1,Train1,Train1,Train1,Train1,Train1]
 - t = Train1
43      10      TP1 @ (1:Queue_System)
 - t = Train3
44      10      Comm @ (1:Queue_System)
 - n =
[Train2,Train2,Train2,Train2,Train2,Train2,Train2,Train1,Train1,Train1,Trai
n1,Train1,Train1,Train1,Train1,Train1,Train1]
 - t = Train2
45      10      Queue @ (1:Queue_System)
 - n =
[Train2,Train2,Train2,Train2,Train2,Train2,Train2,Train1,Train1,Train1,Trai
n1,Train1,Train1,Train1,Train1,Train1,Train1]
 - t = Train3
46      10      TP1 @ (1:Queue_System)
 - t = Train3
47      10      Queue @ (1:Queue_System)
 - n =
[Train2,Train2,Train2,Train2,Train2,Train2,Train2,Train1,Train1,Train1,Trai
n1,Train1,Train1,Train1,Train1,Train1,Train1,Train3]
 - t = Train3
48      10      TP1 @ (1:Queue_System)
 - t = Train3
49      10      Queue @ (1:Queue_System)
```

```
 - n =
[Train2,Train2,Train2,Train2,Train2,Train2,Train2,Train1,Train1,Train1,Trai
n1,Train1,Train1,Train1,Train1,Train1,Train1,Train3,Train3]
 - t = Train3
50      10      TP1 @ (1:Queue_System)
 - t = Train3
51      10      TP1 @ (1:Queue_System)
 - t = Train3
52      10      TP1 @ (1:Queue_System)
 - t = Train3
53      10      TP1 @ (1:Queue_System)
 - t = Train3
54      10      Queue @ (1:Queue_System)
 - n =
[Train2,Train2,Train2,Train2,Train2,Train2,Train2,Train1,Train1,Train1,Trai
n1,Train1,Train1,Train1,Train1,Train1,Train1,Train3,Train3,Train3]
 - t = Train3
55      10      Queue @ (1:Queue_System)
 - n =
[Train2,Train2,Train2,Train2,Train2,Train2,Train2,Train1,Train1,Train1,Trai
n1,Train1,Train1,Train1,Train1,Train1,Train1,Train3,Train3,Train3,Train3]
 - t = Train3
56      10      TP1 @ (1:Queue_System)
 - t = Train3
57      10      Queue @ (1:Queue_System)
 - n =
[Train2,Train2,Train2,Train2,Train2,Train2,Train2,Train1,Train1,Train1,Trai
n1,Train1,Train1,Train1,Train1,Train1,Train1,Train3,Train3,Train3,Train3,Tr
ain3]
 - t = Train3
58      10      TP1 @ (1:Queue_System)
 - t = Train3
59      10      Queue @ (1:Queue_System)
 - n =
[Train2,Train2,Train2,Train2,Train2,Train2,Train2,Train1,Train1,Train1,Trai
n1,Train1,Train1,Train1,Train1,Train1,Train1,Train3,Train3,Train3,Train3,Tr
ain3,Train3]
 - t = Train3
60      10      Queue @ (1:Queue_System)
 - n =
[Train2,Train2,Train2,Train2,Train2,Train2,Train2,Train1,Train1,Train1,Trai
n1,Train1,Train1,Train1,Train1,Train1,Train1,Train3,Train3,Train3,Train3,Tr
ain3,Train3,Train3]
 - t = Train3
61      10      TP1 @ (1:Queue_System)
 - t = Train3
62      10      Queue @ (1:Queue_System)
 - n =
[Train2,Train2,Train2,Train2,Train2,Train2,Train2,Train1,Train1,Train1,Trai
n1,Train1,Train1,Train1,Train1,Train1,Train1,Train3,Train3,Train3,Train3,Tr
ain3,Train3,Train3,Train3]
 - t = Train3
63      10      Queue @ (1:Queue_System)
 - n =
[Train2,Train2,Train2,Train2,Train2,Train2,Train2,Train1,Train1,Train1,Trai
n1,Train1,Train1,Train1,Train1,Train1,Train1,Train3,Train3,Train3,Train3,Tr
ain3,Train3,Train3,Train3,Train3]
 - t = Train3
64      15      Comm @ (1:Queue_System)
 - n =
[Train2,Train2,Train2,Train2,Train2,Train2,Train1,Train1,Train1,Train1,Trai
```

```
n1,Train1,Train1,Train1,Train1,Train1,Train3,Train3,Train3,Train3,Train3,Tr
ain3,Train3,Train3,Train3,Train3]
 - t = Train2
65      20      Comm @ (1:Queue_System)
 - n =
[Train2,Train2,Train2,Train2,Train2,Train1,Train1,Train1,Train1,Train1,Trai
n1,Train1,Train1,Train1,Train1,Train3,Train3,Train3,Train3,Train3,Train3,Tr
ain3,Train3,Train3,Train3]
 - t = Train2
66      25      Comm @ (1:Queue_System)
 - n =
[Train2,Train2,Train2,Train2,Train1,Train1,Train1,Train1,Train1,Train1,Trai
n1,Train1,Train1,Train1,Train3,Train3,Train3,Train3,Train3,Train3,Train3,Tr
ain3,Train3,Train3]
 - t = Train2
67      30      Comm @ (1:Queue_System)
 - n =
[Train2,Train2,Train2,Train1,Train1,Train1,Train1,Train1,Train1,Train1,Trai
n1,Train1,Train1,Train3,Train3,Train3,Train3,Train3,Train3,Train3,Train3,Tr
ain3,Train3]
 - t = Train2
68      35      Comm @ (1:Queue_System)
 - n =
[Train2,Train2,Train1,Train1,Train1,Train1,Train1,Train1,Train1,Train1,Trai
n1,Train1,Train3,Train3,Train3,Train3,Train3,Train3,Train3,Train3,Train3,Tr
ain3]
 - t = Train2
69      40      Comm @ (1:Queue_System)
 - n =
[Train2,Train1,Train1,Train1,Train1,Train1,Train1,Train1,Train1,Train1,Trai
n1,Train3,Train3,Train3,Train3,Train3,Train3,Train3,Train3,Train3,Train3]
 - t = Train2
70      45      Comm @ (1:Queue_System)
 - n =
[Train1,Train1,Train1,Train1,Train1,Train1,Train1,Train1,Train1,Train1,Trai
n3,Train3,Train3,Train3,Train3,Train3,Train3,Train3,Train3,Train3]
 - t = Train2
71      50      Comm @ (1:Queue_System)
 - n =
[Train1,Train1,Train1,Train1,Train1,Train1,Train1,Train1,Train1,Train3,Trai
n3,Train3,Train3,Train3,Train3,Train3,Train3,Train3,Train3,Train3]
 - t = Train1
72      55      Comm @ (1:Queue_System)
 - n =
[Train1,Train1,Train1,Train1,Train1,Train1,Train1,Train1,Train3,Train3,Trai
n3,Train3,Train3,Train3,Train3,Train3,Train3,Train3,Train3]
 - t = Train1
73      60      Comm @ (1:Queue_System)
 - n =
[Train1,Train1,Train1,Train1,Train1,Train1,Train1,Train3,Train3,Train3,Trai
n3,Train3,Train3,Train3,Train3,Train3,Train3,Train3]
 - t = Train1
74      65      Comm @ (1:Queue_System)
 - n =
[Train1,Train1,Train1,Train1,Train1,Train1,Train3,Train3,Train3,Train3,Trai
n3,Train3,Train3,Train3,Train3,Train3,Train3]
 - t = Train1
75      70      Comm @ (1:Queue_System)
 - n =
[Train1,Train1,Train1,Train1,Train1,Train3,Train3,Train3,Train3,Train3,Trai
n3,Train3,Train3,Train3,Train3,Train3]
```

```
 - t = Train1
76      75      Comm @ (1:Queue_System)
 - n =
[Train1,Train1,Train1,Train1,Train3,Train3,Train3,Train3,Train3,Train3,Trai
n3,Train3,Train3,Train3]
 - t = Train1
77      80      Comm @ (1:Queue_System)
 - n =
[Train1,Train1,Train1,Train3,Train3,Train3,Train3,Train3,Train3,Train3,Trai
n3,Train3,Train3]
 - t = Train1
78      85      Comm @ (1:Queue_System)
 - n =
[Train1,Train1,Train3,Train3,Train3,Train3,Train3,Train3,Train3,Train3,Trai
n3,Train3]
 - t = Train1
79      90      Comm @ (1:Queue_System)
 - n =
[Train1,Train3,Train3,Train3,Train3,Train3,Train3,Train3,Train3,Train3,Trai
n3]
 - t = Train1
80      95      Comm @ (1:Queue_System)
 - n =
[Train3,Train3,Train3,Train3,Train3,Train3,Train3,Train3,Train3,Train3]
 - t = Train1
81      100     Comm @ (1:Queue_System)
 - n = [Train3,Train3,Train3,Train3,Train3,Train3,Train3,Train3,Train3]
 - t = Train3
82      105     Comm @ (1:Queue_System)
 - n = [Train3,Train3,Train3,Train3,Train3,Train3,Train3,Train3]
 - t = Train3
83      110     Comm @ (1:Queue_System)
 - n = [Train3,Train3,Train3,Train3,Train3,Train3,Train3]
 - t = Train3
84      115     Comm @ (1:Queue_System)
 - n = [Train3,Train3,Train3,Train3,Train3,Train3]
 - t = Train3
85      120     Comm @ (1:Queue_System)
 - n = [Train3,Train3,Train3,Train3,Train3]
 - t = Train3
86      125     Comm @ (1:Queue_System)
 - n = [Train3,Train3,Train3,Train3]
 - t = Train3
87      130     Comm @ (1:Queue_System)
 - n = [Train3,Train3,Train3]
 - t = Train3
88      135     Comm @ (1:Queue_System)
 - n = [Train3,Train3]
 - t = Train3
89      140     Comm @ (1:Queue_System)
 - n = [Train3]
 - t = Train3
90      145     Comm @ (1:Queue_System)
 - n = []
 - t = Train3
```

World Academy of Science, Engineering and Technology
International Journal of Industrial and Manufacturing Engineering
Vol:10, No:5, 2016

Adopting Flocks of Birds Approach to Predator for Anomalies Detection on Industrial Control Systems

M. Okeke, A. Blyth

***Abstract*—**Industrial Control Systems (ICS) such as Supervisory Control And Data Acquisition (SCADA) can be seen in many different critical infrastructures, from nuclear management to utility, medical equipment, power, waste and engine management on ships and planes. The role SCADA plays in critical infrastructure has resulted in a call to secure them. Many lives depend on it for daily activities and the attack vectors are becoming more sophisticated. Hence, the security of ICS is vital as malfunction of it might result in huge risk. This paper describes how the application of Prey Predator (PP) approach in flocks of birds could enhance the detection of malicious activities on ICS. The PP approach explains how these animals in groups or flocks detect predators by following some simple rules. They are not necessarily very intelligent animals but their approach in solving complex issues such as detection through corporation, coordination and communication worth emulating. This paper will emulate flocking behavior seen in birds in detecting predators. The PP approach will adopt six nearest bird approach in detecting any predator. Their local and global bests are based on the individual detection as well as group detection. The PP algorithm was designed following MapReduce methodology that follows a Split Detection Convergence (SDC) approach.

***Keywords*—**Industrial control systems, prey predator, SCADA, SDC.

I. Introduction

SCADA systems have evolved from single, monolithic entities to the Internet of Things (IoT) [1]. SCADA is used in many different critical infrastructures, from nuclear management to utility, power, waste and engine management on ships and planes. Security of ICS is paramount as deviation from normal operation may result in putting lives at risk. The interconnection of these devices in a distributed environment through the Internet and other means exposes them to various attacks.

The possibility of malicious intent in such an ecosystem of interconnected devices is high. Managing and securing such an ecosystem can be daunting for security engineers and operators due to the dispersed nature of it. Bruce Schneier [2] pointed out that sometimes it seems as if the attackers have the upper hand due to the fact that the technological advancement is faster than security. This is true, as security personnel needs to study the new technology before developing new methods and approaches of securing it.

M. Okeke is a researcher at the University of South Wales, Member of Information Security Research Group, Department of Computing Engineering and Science (e-mail: michael.okeke1@southwales.ac.uk).

A. J. Blyth is the Head of Information Security Research Group, Department of Computing Engineering and Science (e-mail: andrew.blyth@southwales.ac.uk).

ICS generates enormous amount of data that makes it difficult for traditional IDS to analyse. Thus, the volume of data as well as the attack vectors is overwhelming the current detection mechanisms such as witnessed in Stuxnet attack in 2010 [3]. Big Data generators, such as ICS, require new technologies for anomaly detection as well as data processing and storage [4]. Hence, this study presented the use of PP approach seen in flocks of birds in securing ICS.

The rest of this paper is organized as follows: Section II presents the related work with regards to application of PP approach in IDS. Section III justifies the reason for adopting the PP approach while Section IV is the application of the approach. Section V is the introduction of ICSs and Hadoop framework. Section VI is the algorithm design approach by following MapReduce methodology.

II. Related Works

The application of PP approach in IDS is very new. Hence this paper will explore into some areas where the approach has been applied. The PP approach has not been applied in IDS for the protection of ICS. However, research into prey-predator approach has been explored over the years in other areas. Researchers on this have been researching on issues such as fishery population control as well as population dynamics [5]-[7]. The population dynamics in this regard refers to increase or decrease in population of a particular species as a result of certain conditions such as more predators might cause less prey and less prey might bring about death of predators as a result of hunger.

III. Reasons for PP Approach

The idea of using PP approach or defensive mechanism seen in some bird species such as starlings has not been really exploited in IDS. Starlings as preys have their ways of detecting single or multiple predatory attacks through their movements and actions. Their movements and actions are coordinated in such a way that they watch each other's action and this confuses the predator, which is seen as anti-predatory approach named "Confusion Effect" [8]. These birds look alike that the predator finds it difficult to pick on one in their mist and through the movement, the predator is confused, thus making it hard for the predator to catch one. Some hypotheses such as "Many-eyes hypotheses, Chorus line hypotheses" were introduced as a result of birds' collective behaviours [8]. Birds such as Starlings in the group do not have a leader but their behaviours are collective. This collective behavior can be observed in the way they flee the scene when predators are detected. Application of this detection approach in computer

World Academy of Science, Engineering and Technology
International Journal of Computer and Information Engineering
Vol:10, No:7, 2016

[Keynote]: No-Trust-Zone Architecture for Securing Supervisory Control and Data Acquisition

Authors : Michael Okeke, Andrew Blyth

Abstract : Supervisory Control And Data Acquisition (SCADA) as the state of the art Industrial Control Systems (ICS) are used in many different critical infrastructures, from smart home to energy systems and from locomotives train system to planes. Security of SCADA systems is vital since many lives depend on it for daily activities and deviation from normal operation could be disastrous to the environment as well as lives. This paper describes how No-Trust-Zone (NTZ) architecture could be incorporated into SCADA Systems in order to reduce the chances of malicious intent. The architecture is made up of two distinctive parts which are; the field devices such as; sensors, PLCs pumps, and actuators. The second part of the architecture is designed following lambda architecture, which is made up of a detection algorithm based on Particle Swarm Optimization (PSO) and Hadoop framework for data processing and storage. Apache Spark will be a part of the lambda architecture for real-time analysis of packets for anomalies detection.

Keywords : industrial control system (ics, no-trust-zone (ntz), particle swarm optimisation (pso), supervisory control and data acquisition (scada), swarm intelligence (SI)
Conference Title : ICCCIS 2016 : 18th International Conference on Cryptography, Coding and Information Security
Conference Location : London, United Kingdom
Conference Dates : July 28-29, 2016

2017 IEEE 3rd International Conference on Electro-Technology for National Development (NIGERCON)

Emulating the Distributed Detection Approach in Flocks of Birds for Securing SCADA Systems

Michael Okeke
Information Security Research Group
University of South Wales
Pontypridd, United Kingdom
michael.okeke1@southwales.ac.uk

Andrew Blyth
Information Security Research Group
University of South Wales
Pontypridd, United Kingdom
Andrew.blyth@southwales.ac.uk

***Abstract*— European Starlings and other bird's flocks in thousands and even millions. Their flocking behaviors attract predators that see such collective behavior as an easy target. However, the surprising thing is that it is harder to target and capture one in their mist than a lone bird. It is easy to target and capture a lone bird than a bird in a flock. One of these collective behaviors is the capability of each of them to detect a predator. Each one in the flock can detect a predator some meters away. The more they are in number, the harder it will be not to detect the incoming predator. This paper proposed modelling the same approach for securing Industrial Control System (ICS) and in particular Supervisory Control and Data Acquisition (SCADA). However, this approach seen in the flock of birds can be replicated in computer system for securing it. Train system was used as a testbed in this paper for data collection. Live data from the train system was collected, analyzed and some risk points were discovered. This risk points were the entry points for anomalies into the system. These are the points that needs securing. The model was to show the viability of using the approach as well as integrating it on SCADA systems, and the train system was a representation of SCADA system. This approach is purely distributed in the sense that each bird is capable of detecting a predator. The detection of anomalies is based on the location of each bird in the system. The bird's environment has cells and each cell has number. The maximum number of birds each cell can contain was six. Six birds approach was used for the detection and this can be seen in the model. Whenever a packet arrives, the first cell that contain six birds will be activated for detection. Birds in the cell has numbers and the activity is based on the number. The results showed that both known and unknow anomalies can be detected on SCADA systems.**

***Keywords*— Detection; Flocking; Industrial Control System (ICS); Model; SCADA; Security.**

I. Introduction

The rate at which technology is advancing has posed too many questions with regards to securing those technologies against malicious intent. This was also evident in the recent development in the insulin pump that uses wireless connection. The pump comes with the possibility of adjusting the level of insulin remotely, which poses a lot security issues as stated in [1]. These are all control system that is built to make life easier but the enemy can use the advantage if left unattended or unsecure.

The attack on the Iranian SCADA system was one of the wakeup call to protect the process control system. The virus was labeled Stuxnet, which penetrated the system through the vendors Driver DLL. The Driver DLL was infected which gave the attacker the privilege of penetrating the control system and multiplying, infecting and causing damages to the system [2]. The whole system was affected through a USB stick that was used to download the driver. This shows that the attack vectors are changing following the sophistication of the attack on the Iranian plant and other recent attacks. As Langner once said that the process of securing the systems needs to be stepped up. We have witnessed some train crash and collisions in recent times in Spain, Germany and Italy [3]-[5].

The idea of using the detection approach seen in the flocks of bird for securing ICS was first proposed by [6]. However, their proposed idea was an introduction to the possibility of using the collective behavior seen in the flocks of bird. Their work did not produce any model at the time, hence this work is an extension of the work. This research will produce a model that can be integrated into the SCADA system for the detection of anomalies. The work will produce a model as an approach on how detection will be carried out in the SCADA environment by following the flocks of bird's approach. Real data from SCADA testbed will be used to determine the areas of anomalies as well as the model integration point.

This paper proposed emulating the detection approach seen in the flocks of birds for Intrusion Detection System (IDS) for protecting the Supervisory Control and Data Acquisition (SCADA). The approach was modelled using Unified Modelling Language (UML). The model will help in explaining the behavior that is essential in detecting anomalies on ICS. The data collected from the SCADA testbed was used to determine the structure and other risks associated with SCADA system. Hence, the evidence shows that the possibilities of this data being contaminated are high as